UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA TÉCNICA EN INFORMÁTICA DE GESTIÓN

PROYECTO FIN DE CARRERA

Análisis, diseño e implementación de mejoras en el servicio de sellado espacio-temporal de CERTILOC

AUTOR: Raúl David Martínez Calmaestra
TUTOR: Dra. Dña. Ana Isabel González-Tablas Ferreres

TÍTULO: Análisis, diseño e implementación de mejoras en el servicio de sellado espacio-temporal de CERTILOC

AUTOR: D. RAÚL DAVID MARTÍNEZ CALMAESTRA

TUTORA: DRA. DÑA. ANA ISABEL GONZÁLEZ-TABLAS FERRERES

La defensa del presente Proyecto de Fin de Carrera se realizó el día 24 de Octubre de 2012; siendo calificada por el siguiente tribunal:

Presidente: José Daniel García Sánchez

Secretario: J.M de Fuentes García-Romero de Tejada

Vocal: Jorge Blasco Alís

Habiendo obtenido la siguiente calificación: Sobresaliente Matrícula de Honor

Presidente	Secretario	Vocal
José Daniel García Sánchez	J.M de Fuentes García-Romero de Tejada	Jorge Blasco Alís

AGRADECIMIENTOS

A mi esposa Carolina, por su apoyo incondicional día tras día, hora tras hora y minuto a minuto durante todos estos meses que he dedicado a “proyectar”.

A mi tutora Anabel, por la oportunidad que me ha dado de realizar este proyecto y por toda la ayuda que me ha brindado.

A todos aquellos familiares y amigos que me ha dado alguna palabra de ánimo y cariño durante estos meses.

A Sir Clive Marles Sinclair, por inventar el ordenador personal Sinclair ZX Spectrum 48Kb.

RESUMEN

El presente documento describe el procedimiento seguido para la realización del Proyecto de Fin de Carrera, desarrollado por el alumno Raúl David Martínez Calmaestra, para la obtención del título de Ingeniería Técnica en Informática de Gestión.

El proyecto desarrollado se engloba dentro del contexto del proyecto de investigación CERTILOC. Dicho proyecto tiene como objetivo general el proporcionar un marco de seguridad y legalidad para varios métodos y tecnologías de localización espacio-temporal existentes hoy en día.

Particularmente, los módulos desarrollados en el proyecto que presentamos a continuación, aportan a CERTILOC una serie de extensiones funcionales y técnicas sobre la implementación diseñada y desarrollada en el ámbito del Proyecto de Fin de Carrera "Implementación de protocolos de sellado temporal y sellado espacio-temporal en XML para CERTILOC". La implementación realizada en dicho proyecto sigue el Protocolo González-Tablas et al. (2005), que provee servicios SSET (Sistema de Sellado Espacio Temporal).

Estos servicios SSET se componen de lo siguiente:

- Un servicio de acreditación /certificación espacio-temporal (SAET).

- Un servicio de sellado temporal (SST) que proporciona evidencias sobre la existencia de la firma digital antes de un determinado instante de tiempo.

El software implementado en el presente proyecto presenta diversas modificaciones funcionales y técnicas que se pueden resumir en las principales:

- Implementación de interfaz de usuario cómoda para el usuario, de manera que el usuario tenga una visión clara de la gama de posibles acciones a realizar mediante este software y qué acción se está ejecutando en cada momento.

- Implementación de una arquitectura cliente servidor para adecuarse a un entorno real en el que cada servicio (SAET,SST) está provisto por una máquina servidora distinta.

- Obtención del tiempo de una fuente confiable de tiempo (servicio SST) para asegurar las condiciones temporales del Sello Espacio Temporal. Estas fuentes confiables de tiempo pueden ser servidores que sigan el protocolo NTP, por ejemplo.

- Mejoras en la validación de firmas y certificados digitales (validación de distintas firmas de distintos documentos xml, validación de rutas de certificación complejas)

- Mejoras internas de código, modularización, arquitectura, escalabilidad, etcétera.

ÍNDICE GENERAL

LISTA DE FIGURAS

LISTA DE TABLAS

GLOSARIO DE TÉRMINOS

Por motivos de espacio o legibilidad a veces denominamos a la misma cosa tanto en la forma normal como en forma de abreviatura, es por eso que incluímos aquí los términos más comunes:

TÉRMINO	TÉRMINO EN INGLÉS	SIGNIFICADO
M		A menudo en algunos diagramas, abreviatura del Mensaje que se quiere sellar Espacio-Temporalmente
CET	STC (Spatial-Temporal Certificate)	Abreviatura de Certificado Espacio Temporal, documento que emite CERTILOC como parte del servicio de acreditación espacio temporal. Dicho documento acredita que un dispositivo, y por extensión la persona que lo posee, están en una determinada localización espacial y temporal.
SET	STS (Spatial-Temporal Stamp)	Documento final que provee el software presentado en el presente proyecto, que forma parte de un servicio de acreditación espacio temporal. Acredita que cierto documento M existe en ciertas condiciones de espacio y tiempo.
NTP		Network Time Protocol, protocolo de obtención de fuente de tiempo confiable.
Sigma		Documento que por parte del Sujeto se prepara y que contiene un resumen tanto del Mensaje como del CET, y el conjunto firmado con la clave privada del Sujeto

1. INTRODUCCIÓN

El presente documento describe el proceso de realización del Proyecto de Fin de Carrera (PFC) desarrollado por el autor para la obtención del título de Ingeniero Técnico en Informática de Gestión.

Este proyecto consistirá en correcciones y mejoras funcionales y técnicas sobre la implementación realizada en el Proyecto de Fin de Carrera "Implementación de protocolos de sellado temporal y sellado espacio-temporal en XML para CERTILOC", de Alvaro Gascón, 2008 ([1]).

Dicho proyecto consiste en diseñar e implementar un sistema acorde con el modelo de sellado espacio-temporal definido en el seno del proyecto de investigación CERTILOC. En las siguientes secciones explicaremos en qué consiste CERTILOC, cual es la problemática a tratar en el presente proyecto y cómo está estructurada la memoria.

1.1 Contexto: el Proyecto CERTILOC

Importancia de los nuevos sistemas de ubicación geográfica

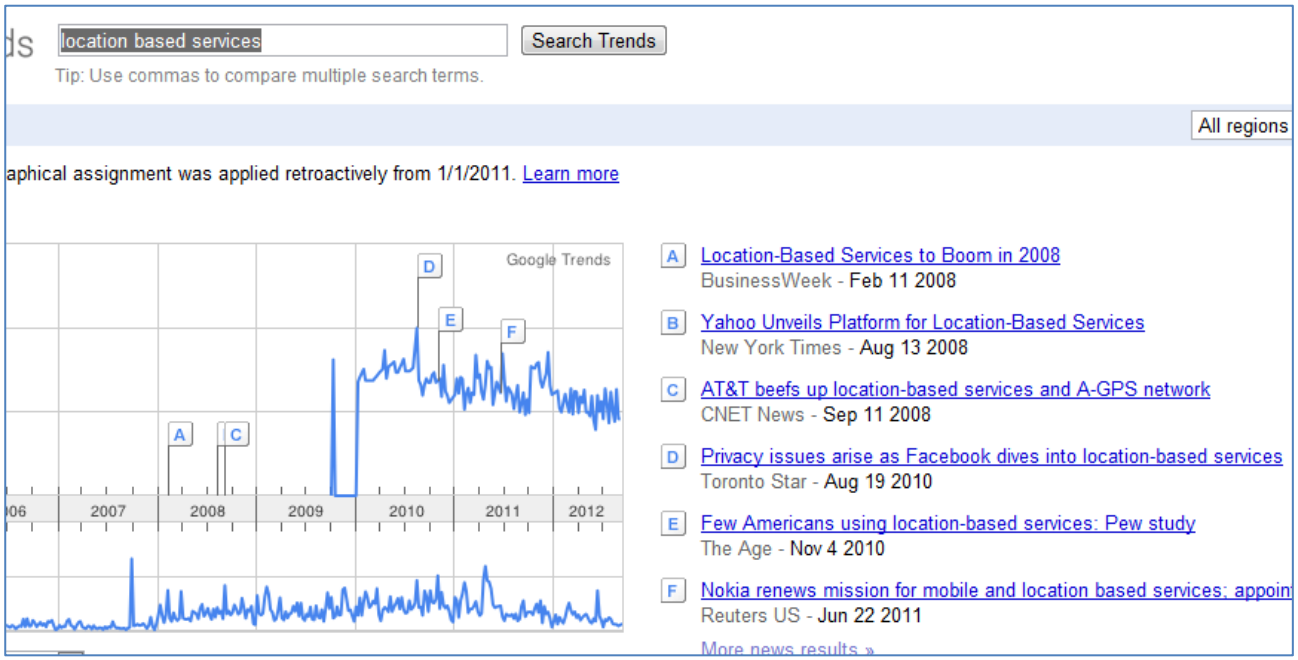
En los últimos años es sabido que se ha vivido un auge de los teléfonos móviles entre la población mundial. Una de los primeros potenciales descubiertos en esta tecnología es en el ámbito de la ubicación geográfica de teléfonos móviles. Antes de la implantación de las tecnologías de localización por satélite (GPS) se podía localizar un teléfono, y por lo tanto a su poseedor, a través de lo que se conoce como "localización GSM", que es un servicio ofrecido por las empresas operadoras de telefonía móvil que permite determinar, con una cierta precisión, donde se encuentra físicamente un terminal móvil determinado porque mediante técnicas de triangulación se puede calcular en las cercanías de qué célula se encuentra ubicado.

Más recientemente[1] se ha producido un crecimiento en las ventas de dispositivos móviles dotados de sensores de ubicación geográfica mediante satélite (GPS). Tambien se han popularizado otros dispositivos como tabletas (tablets) e incluso en los últimos años existe la posibilidad de tener un ordenador personal, ya sea de escritorio o portátil, conectado a una red de telefonía móvil por medio de los nuevos dispositivos 3G. Tambien hay Sistemas de Posicionamiento Híbrido ofrecidos por diversas empresas que aportan el valor añadido de solventar el problema de cobertura GPS en interiores solventándolo con otras técnicas.

La nueva generación de dispositivos móviles ha visto incrementadas notablemente su potencia técnica, no solo por las nuevas tecnologías hardware que portan (Wi-fi, bluetooth, GPS) sino también a nivel de potencia de proceso, memoria RAM, etc. A nivel de software, esta nueva generación de dispositivos móviles inteligentes permite a los desarrolladores de software mayores capacidades para programar aplicaciones que aprovechen las características con las que vienen dotados dichos dispositivos y es aquí donde tenemos que hacer mención a los Servicios Basados en Localización (Location Based Services, LCS) que son aquellos servicios que se basan en información de localización para aportar un valor añadido y del que disponemos de un amplio abanico de posibles aplicaciones relacionadas con las capacidades de ubicación geográfica como sistemas de navegación vehicular, información de tráfico, turística, páginas amarillas geográficas (como *Google Maps*) e incluso redes sociales que permiten localizar amistades como *Foursquare* o *Facebook Lugares*.

[1] *El teléfono Nokia 95 fue el primer teléfono de difusión mundial en venderse dotado con GPS, en Abril de 2007.*

La siguiente figura, resultado de una búsqueda en la herramienta Google Trends, ilustra la tendencia de interés en estos llamados servicios basados en localización:



Como puede apreciarse, antes de 2009 apenas se hablaba del tema, pero a finales del año 2009 se disparó el interés por los mencionados servicios.

El proyecto CERTILOC y por ende este proyecto de fin de carrera están relacionados con esta difusión de las tecnologías de ubicación geográfica porque junto a estas tecnologías se hace necesario implementar una serie de servicios de seguridad de la información de vital importancia debido a la naturaleza sensible de esta información geográfica y a las posibilidades de notarizar electrónicamente la exactitud de dicha información.

El proyecto CERTILOC

El presente proyecto se enmarca dentro del proyecto de investigación CERTILOC, que se está llevando a cabo por el Grupo de Seguridad de la Información. Dicho proyecto consiste en definir la metodología, protocolos e implementación de un sistema de localización espacio temporal que además provea servicios de seguridad de información.

Es en este contexto donde se ha introducido el concepto de Información Espacio Temporal (IET). LA IET es la información que indica la posición de una entidad en una dimensión geográfica y en una dimensión temporal. La ubicación geográfica o de localización se puede proporcionar mediante Servicios de Localización (Location Services, aquellos que permiten determinar la información geográfica de un dispositivo móvil y por lo tanto, del individuo asociado a este) o bien mediante Servicios Basados en Localización (Location Based Services, LCS) que son aquellos servicios que se basan en información de localización para aportar un valor añadido, tal es el ejemplo que hemos citado en el apartado anterior antes de las aplicaciones móviles de nueva generación.

En el ámbito del proyecto CERTILOC se encuentra una necesidad de aportar seguridad de la información geográfica y temporal asociada a un dispositivo porque esta a su vez proporciona información del individuo portador del mismo y esta información es de naturaleza sensible, y también pero no menos importante tenemos que tener un servicio que nos permita demostrar legalmente la exactitud de dicha información.

Esta seguridad de la información se proporciona mediante unos servicios de seguridad como se explica en [1] y que resumimos a continuación:

- *Servicios de confianza: son servicios prestados por un Tercero de Confianza, que es una entidad que puede hacerse responsable de una determinada información. Estos se dividen en:*

    - *Servicios de autenticación, que permiten verificar la identidad de una entidad.*

    - *Servicios de notarización/certificación: Son servicios en los que un Tercero de Confianza da fe de la veracidad y exactitud de una serie de datos mediante la generación de evidencias con capacidad probatoria.*

      Dentro ellos se enmarcan los Servicios de Acreditación y Sellado Espacio Temporal (SASET), que se dividen en Servicios de acreditación espacio temporal (SAET) y Servicios de Sellado Espacio Temporal (SSET) y de los que hablaremos a continuación puesto que son los objetivos principales del Demostrador de CERTILOC.

- *Servicios de privacidad: Según [12] La Ley Orgánica 15/1999 de 13 de diciembre de Protección de Datos de Carácter Personal, (LOPD), es una Ley Orgánica española que tiene por objeto garantizar y proteger, en lo que concierne al tratamiento de los datos personales, las libertades públicas y los derechos fundamentales de las personas físicas. Su objetivo principal es regular el tratamiento de los datos y ficheros, de carácter personal, independientemente del soporte en el cual sean tratados, los derechos de los ciudadanos sobre ellos y las obligaciones de aquellos que los crean o tratan.*

  En el contexto del proyecto CERTILOC, este carácter de privacidad se aplica en que la información de localización no debe poder ser accedida por entidades no autorizadas y la identidad del usuario del dispositivo no tiene por qué ser conocida por la entidad que presta el servicio.

Como se también se explica en [1], *los objetivos del proyecto de investigación CERTILOC son, en primer lugar plantear un modelo teórico que permita certificar la información espacio temporal de una determinada entidad (esta área esta cubierta por la tesis expuesta en [2]) y en segundo lugar implementar un demostrador que nos permita verificar y probar el funcionamiento de dicho modelo. Dicho demostrador ha sido desarrollado conjuntamente en una serie de proyectos ([3], [4] y [5]) y provee una serie de servicios de seguridad que en conjunto se denominan Servicios de Acreditación y Sellado Espacio Temporal (SASET):*

- *Servicio de acreditación Espacio-Temporal (SAET), cuyo objetivo es acreditar las condiciones espacio-temporales de una entidad o sujeto S de las evidencias. Las evidencias que emiten este tipo de servicios reciben el nombre de credenciales o Certificados Espacio Temporales (CET) y se denotarán en este PFC como Θ.* En este sentido, el demostrador tiene las funcionalidades de emisión de CET, control del ciclo de vida del CET.
- *Servicio de Sellado Espacio Temporal (SSET), cuyo objetivo es acreditar que un determinado documento (M) existía en un lugar determinado en cierto momento temporal o que un sujeto S realizó determinada acción sobre dicho documento bajo ciertas condiciones espacio-temporales. En este caso, las evidencias se denominan sellos espacios-temporales (SET) y serán denotados en este PFC por ψ.* En este área se enmarca la implementación el proyecto [1] en la que se basa el presente proyecto.
- *Servicio de Provisión de Privacidad del Usuario.* En este sentido el demostrador tiene implementadas una serie de políticas de privacidad.

1.2 Descripción del problema

Los módulos desarrollados en el proyecto que presentamos a continuación, aportan a CERTILOC una serie de extensiones funcionales y técnicas sobre la implementación diseñada y desarrollada en el ámbito del Proyecto de Fin de Carrera [1]. "Implementación de protocolos de sellado temporal y sellado espacio-temporal en XML para CERTILOC". La implementación realizada en dicho proyecto sigue el Protocolo especificado en [2] González-Tablas et al. (2005), que provee servicios SSET (Sistema de Sellado Espacio Temporal).

Dicha implementación adolece de una serie de carencias que hemos identificado y enumerado en detalle en la sección de esta memoria 4. ANÁLISIS DEL PROBLEMA Y ALTERNATIVAS DE SOLUCIÓN y que podemos considerar como nuestro Catálogo de Requisitos en el ámbito de realización del presente proyecto software.

Dichas carencias son de tres tipos según su origen: a) Líneas futuras a seguir como se enumera en la sección correspondiente de la memoria de [1], b) Carencias funcionales y c) Carencias técnicas.

Las más importantes de estas carencias a cubrir en el presente proyecto son:

- **Falta de usabilidad del software implementado** al estar definida como una serie de comandos que hay que ejecutar en un orden concreto, y que además no permiten al usuario seleccionar cuáles van a ser las entidades Mensaje y CET a utilizar, sino que está todo ya establecido "de fábrica".

- **Falta de confianza en la fuente de información temporal** que utiliza el Servicio de Sellado de Tiempo (SST). Dicha información se obtiene de la hora del sistema, y ésta puede estar sujeta a errores ya que depende de factores externos al software implementado que no proveen de confianza per se. Por ejemplo, la hora del sistema puede estar desfasada con respecto a la zona horaria correcta, horario de verano, reseteada por reinstalaciones de equipo, caídas de corriente, manipulada malintencionadamente, etc.

- **Falta de semántica del formato de XML propuesto** para el Sello Espacio Temporal, porque no hay un enlace claro entre dicho documento con las entidades Mensaje y CET, que forman parte del mismo. Por otro lado, la manera de implementar el nodo MessageImprints es incorrecta, pues no tiene en cuenta el uso del nodo tsp:References (Más información en la sección 5.4)

- **Falta de adecuación a un entorno real** porque todo el software está hecho para ser ejecutado en una misma máquina. En un entorno real el dispositivo móvil debería ser un ente independiente físicamente de la arquitectura software que implemente los Servicios de Acreditación y Sellado Temporal.

- **Falta de comprobación de validez de la información de clave pública**. No se comprueba que los certificados de clave pública incluídos en las firmas XML Signature son válidos.

- Relacionado con el punto anterior, **insuficiente soporte para las credenciales de firma** de las distintas entidades del protocolo. La implementación solamente está preparada para un certificado de clave pública por cada firma XML Signature, cuando en un entorno real la ruta de certificación puede ser más compleja (varios certificados de clave pública por cada firma) y se debe incluir la ruta de certificación completa en la firma XML Signature para que esta sea verificable.

- **Necesidad de mejora de la arquitectura del software** implementado. Encapsulamiento de clases incorrecto, falta de modularidad, herencia de clases innecesaria, nula configurabilidad.

1.3 Objetivos del Proyecto de Fin de Carrera

Como se dijo anteriormente, el presente Proyecto de Fin de Carrera se encarga del diseño y desarrollo de las modificaciones necesarias para extender el funcionamiento y resolver las carencias del software ya implementado de un sistema de sellado espacio-temporal en el marco del proyecto CERTILOC.

Todas estas modificaciones pretenden cubrir las carencias explicadas en la sección 1.2 Descripción del problema de la presente memoria.

Para llevar a cabo dicha tarea, se fijan los siguientes objetivos generales:

- Mejorar el XML que representa el Sello Espacio Temporal final para que incluya información más completa sobre el documento sellado y sobre la información de localización.

- Implementación de una interfaz gráfica de usuario (GUI). El sistema debe tener una serie de menus con un abanico de opciones para ejecutar los distintos pasos del protocolo de Sellado Espacio Temporal, el requisito es que esta interfaz sea sencilla e intuitiva

- Confianza en la fuente de información temporal. Debe incluirse en el sistema una fuente confiable de tiempo para la emisión de sellos temporales.

- Implementación de una arquitectura cliente servidor para adecuarse a un entorno real en el que cada servicio (SAET y SST) está provisto por una máquina servidora distinta.

- Validación de la información de clave pública incluída en el XML del Sello Espacio Temporal.

- Realizar las mejoras de arquitectura y código necesarias para que este sea modular y escalable.

1.4 Organización de la memoria

La memoria está estructurada de la siguiente manera:

- **Capítulo 1. Introducción**
  Resumen del contenido del proyecto, destacando los puntos principales, su estructura y sus objetivos, así como una primera aproximación al contenido de esta memoria.

- **Capítulo 2. Gestión del Proyecto**
  Se describen aspectos relacionados con la gestión del proyecto, mostrando, entre otras cosas, la planificación del proyecto, la metodología y las herramientas que se han utilizado. También se incluye un presupuesto del proyecto.

- **Capítulo 3. Estado de la Cuestión**
  Explicación del ámbito de aplicación del presente proyecto en cuanto a los Servicios de Acreditación y Sellado Espacio-Temporal y su relación con el ámbito de la Seguridad Informática

- **Capítulo 4. Análisis del Problema y Alternativas de Solución**
  Relación de propuestas de mejora y propuestas de solución para el software implementado en el Proyecto de Fin de Carrera "Implementación de protocolos de sellado temporal y sellado espacio-temporal en XML para CERTILOC"

- **Capítulo 5. Análisis y Diseño**
  Explicación de la solución propuesta a alto nivel describiendo las modificaciones a realizar sobre documentos, algoritmos, estructuras de certificación y comunicación entre las entidades principales (Sujeto, TSA y CERTILOC)

- **Capítulo 6. Implementación**
  Explicación detallada de la arquitectura de la solución propuesta y de la implementación de las principales clases envueltas en el desarrollo del proyecto.

- **Capítulo 7. Pruebas**
  Explicación detallada de las pruebas realizadas para asegurar la calidad del software y la integridad y no repudio de las distintas firmas digitales involucradas en el servicio de SET implementado.

- **Capítulo 8. Conclusiones y Líneas Futuras**
  Conclusiones obtenidas tras la realización del proyecto y futuras mejoras tanto funcionales como técnicas.

- **Bibliografía**
  Relación de la documentación utilizada para la realización del presente Proyecto.

- **Apéndice A: Manual de Usuario**
  Explicación de cómo utilizar el software implementado tanto para nivel usuario como para usuarios avanzados.

- **Apéndice B: Manual de Instalación**
  Guía de los pasos a seguir para instalar el software implementado.

- **Apéndice C: Manual de Mantenimiento**
  Recomendaciones de configuración, software a instalar, puntos a considerar a la hora de preparar el software implementado para futuros mantenimientos, extensiones y mejoras de funcionalidad.

2. GESTIÓN DEL PROYECTO

2.1. Introducción

En este capítulo se definen todas las actividades de gestión del proyecto. Gracias a él quedarán reflejadas todas las funciones técnicas y de gestión así como las actividades y tareas necesarias para satisfacer los requerimientos del proyecto software. En este capítulo se tratarán los temas relacionados con la panorámica del proyecto, la organización del proyecto, el proceso de gestión, el proceso técnico, la planificación y el presupuesto.

2.1.1. Panorámica del proyecto

El presente proyecto se realiza de acuerdo principalmente siguiendo los mismos requisitos que se especificaron para el proyecto [1], y secundariamente a los requisitos recopilados en la sección 4 Análisis del Problema y Alternativas de Solución de esta memoria.

2.2. Organización del proyecto

2.2.1. Modelo del proceso

El ciclo de vida del proyecto es lo que se denomina en ingeniería del software “Modelo en
Cascada con retroceso”. Este ciclo de vida es lineal, tal y como lo es el ciclo en el que se basa pero con algunas diferencias. Su funcionamiento es el siguiente: las fases de desarrollo son Análisis,
Diseño, Implementación y Pruebas, tal y como refleja el ciclo en Cascada. La forma de trabajar en cada fase del mismo es exactamente igual a la forma de hacerlo en el de Cascada, pero en el caso de fallo, podemos volver siempre a cualquier fase anterior para arreglar el problema, de esta manera no nos cerramos en un ciclo clásico evitando en la medida de lo posible los bloqueos en el desarrollo de la aplicación.

Una posible representación del modelo se puede ver en la figura siguiente:

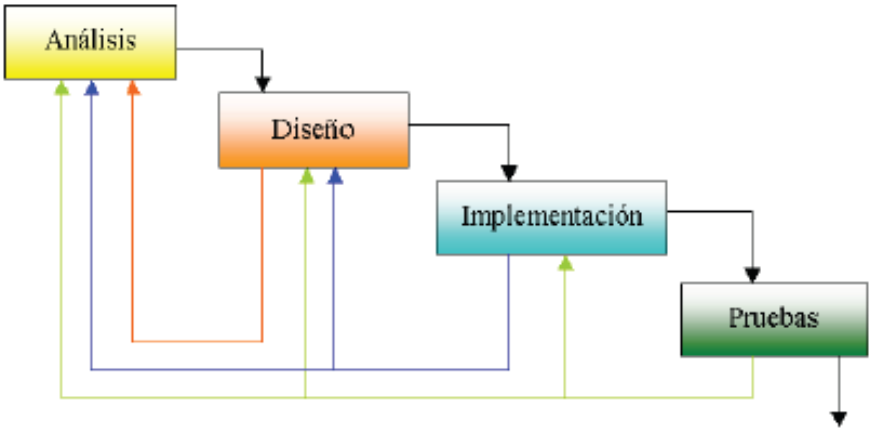


Figura 2.2.1: Ciclo de vida del software

2.2.2. Metodología

A continuación se muestra una descripción de cada una de las fases del modelo:

- Requisitos. En esta fase se elabora una especificación completa y validada de las funciones requeridas, sus interfaces y el rendimiento del producto de software. Los requisitos se encuentran reflejados en el proyecto [1] y en la sección 4 Análisis del Problema y Alternativas de Solución de esta memoria.
- Análisis. En esta fase se crean los modelos estáticos, dinámicos y funcionales, reflejados en diagramas de clases y de secuencia. Se encuentra reflejado en el capítulo 5.
- Diseño. La fase de diseño permite el refinamiento del análisis para dejar el modelo justo a punto para la implementación. Se puede encontrar en el capítulo 5.
- Implementación. En la fase de implementación se desarrolla lo estipulado en el diseño y se construye el sistema que será sometido a pruebas en la fase siguiente. Se encuentra descrita en el capítulo 6.
- Pruebas. El objetivo de esta fase es llevar a cabo el control de fallos y garantizar la calidad del software, basado principalmente en lo producido en implementación. Las diferentes pruebas se encuentran reflejadas en el capítulo 7.

En cualquier momento se puede retroceder a una fase anterior si las circunstancias lo requieren o nuevas funcionalidades son requeridas.

Cabe destacar también que cada una de estas fases se desarrollaran en paralelo con las tareas de:

- Gestión del proyecto
- Verificación y validación
- Formación
- Documentación

2.3. Proceso de gestión del proyecto

En este apartado serán tratados los temas de las prioridades y objetivos de la gestión del proyecto así como la gestión de riesgos y las estrategias para mitigarlos. Un proyecto software son todas las actividades técnicas y de gestión requeridas para proporcionar los entregables al cliente. Un proyecto de software tiene una duración específica, consume recursos y produce productos de trabajo. El proceso de gestión del proyecto software es el destinado a la planificación y organización de los recursos disponibles para controlar y dirigir el proyecto software.

2.3.1. Objetivos de la gestión

Los objetivos de la gestión de este proyecto son:

- Desarrollar una aplicación software que cumpla escrupulosamente con los requisitos establecidos consiguiendo de esta manera un producto de calidad.
- Aprender la utilización de los aspectos tecnológicos y técnicos utilizados en el desarrollo dela aplicación.
- Desarrollar un producto fácilmente mantenible y extensible, de forma que nuevas funcionalidades puedan serle añadidas en el futuro.
- Elaboración de una memoria que explique tanto el aspecto técnico de la aplicación como el aspecto de gestión del proyecto.
- Respetar los plazos de entrega holgadamente.

2.3.2. Gestión de riesgos

Según [14], un riesgo es una valoración subjetiva de la probabilidad de no alcanzar los objetivos deseados en términos de tiempo, coste y recursos asignados. Un riesgo es la posibilidad de sufrir pérdidas durante el ciclo de vida del proyecto. En un proyecto en desarrollo, la pérdida describe su influencia en el proyecto en la forma de calidad disminuida del producto final, costes más elevados, retrasos, o fallos en el proyecto.

Por otro lado, y según la misma fuente, la identificación de los riesgos es un proceso de identificación de los riesgos conocidos y desconocidos del proyecto. La localización del riesgo puede producirse en distintas etapas del proyecto. El responsable aísla las causas de riesgo en el análisis y determina la influencia del riesgo, determinando el tipo de métrica para alto, medio y bajo riesgo. El responsable también computa medidas cuantitativas de probabilidad e influencia en coste, planificación, y rendimiento.

En este caso, el responsable de la gestión de riesgos es el proyectista, que ha establecido un análisis de riesgos siguiendo las pautas anteriores

	Probabilidad de ocurrir	Severidad	Descripción
1	Baja	Baja	Retrasos debidos a falta de formación y experiencia en las tecnologías a utilizar
2	Media	Media	Falta de soporte y de documentación en algunas de las tecnologías utilizadas, debido a que aún se encuentran en desarrollo
3	Alta	Alta	Falta de tiempo para cumplir los plazos debido a otros compromisos laborales

Las estrategias planeadas para mitigar estos riesgos (clasificadas según el modelo anterior) son las siguientes:

Riesgo	Estrategia
1	Obtener libros o tutoriales recomendados por personas con experiencia en las tecnologías utilizadas
2	Utilización de foros de las tecnologías citadas, y preguntas directas a personas con experiencia, siempre que sea posible
3	Aprovechar los periodos de vacaciones en la medida de lo posible

2.3.3. Mecanismos de monitorización y control

Se fijan reuniones semanales con la tutora del proyecto, en su despacho principalmente durante las primeras fases del proyecto y telefónicas / vía e-mail en fases de desarrollo más avanzadas.

2.4. Proceso técnico

En esta sección se detallarán todos los elementos que se utilizarán a lo largo del proceso de desarrollo de la aplicación, abarcando los métodos, herramientas y técnicas utilizadas, así como la documentación generada.

2.4.1. Métodos, herramientas y técnicas

Los distintos recursos utilizados en el desarrollo de este proyecto son:

- Fungibles: Los fungibles utilizados serán básicamente el material normal de oficina (folios, bolígrafos, cuadernos, etc).
- Herramientas hardware
  - Portátil Dell latitude E5410. Intel Core i5 M520 2.4Ghz. 64 bits. 4Gb RAM.
- Herramientas software
  - Entorno: Ubuntu Linux y Microsoft Windows 7 con máquina virtual Windows XP SP2
  - Implementación: JDK 6, Eclipse 4.2
  - Documentación: MS Word 2010
  - Presentaciones: MS Powerpoint 2010
  - Documentación de las clases Java: Se utiliza JavaDoc como medio para documentar las distintas clases Java con el mayor detalle posible y de una forma estructurada.

2.5. Calendario y presupuesto

2.5.1. Calendario

El presente proyecto ha transcurrido desde el 23 de Julio de 2010 hasta el 24 de Octubre de 2012. Para la elaboración del presente proyecto se han seguido las fases que se muestran en la siguiente tabla, con el tiempo que ha requerido cada una de ellas. El motivo de la dilatación en el tiempo de este proyecto es que el proyectista ha tenido que compatibilizarlo con su actual empleo.

Por otro lado, a continuación mostramos el calendario realizado de proyecto. No proporcionamos un calendario de planificación porque se han ido identificando nuevos requisitos en la primera mitad del mismo, lo que ha dificultado planificar los tiempos. Se podrán observar solapes de realización de algunas tareas, porque recordemos que para este proyecto se ha utilizado el "Modelo en Cascada con retroceso", que implica que a menudo volvemos a la tarea anterior en función de las dificultades encontradas en la tarea presente, e incluso a veces adelantamos en la tarea presente algunas subtareas de la tarea posterior, tal es el caso de la tarea de diseño, en la que a menudo se suelen hacer prototipos de la arquitectura a emplear. En el caso de las Pruebas, algunas de ellas son implementaciones software (validaciones de firmas y certificados,etc)

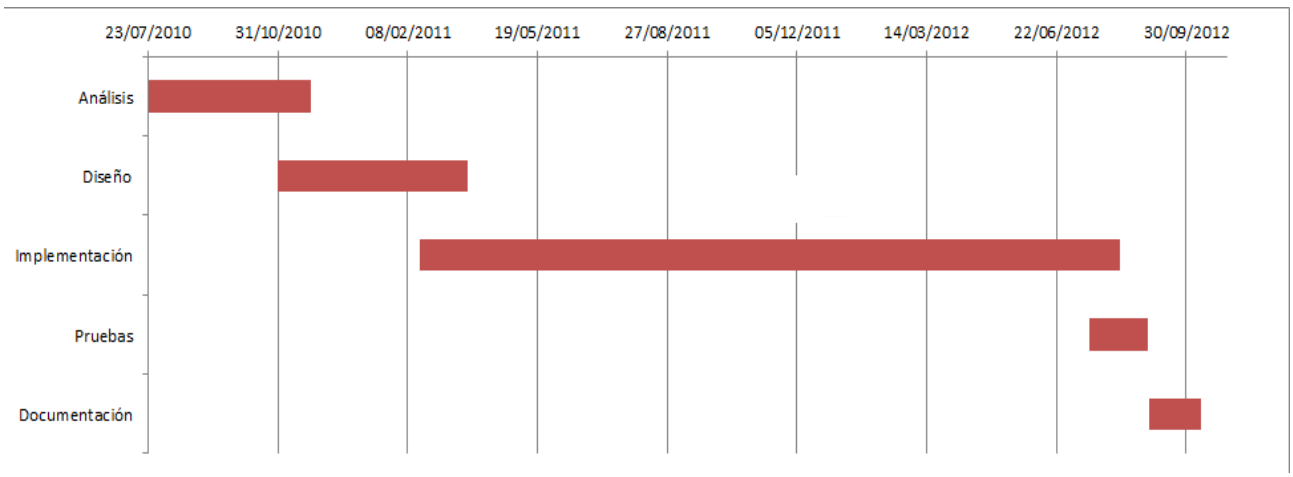


Figura 2.5.1 Calendario de proyecto

Como mencionamos anteriormente, el presente proyecto se ha dilatado 27 meses debido a que el proyectista lo ha tenido que compatibilizar con su actual empleo, sin embargo la siguiente tabla muestra las horas de trabajo efectivo que serán las contabilizadas para el presupuesto:

Tarea	Horas
Análisis	180
Diseño	120
Implementación	780
Pruebas	40
Documentación	150

2.5.2. Presupuesto y gastos

En este apéndice se presentan justificados los costes derivados de la realización del presente proyecto. El presupuesto estará compuesto de los gastos de desarrollo de la aplicación, licencias software, del material fungible necesario y del coste del único equipo informático adquirido (un ordenador portátil).

Desglose de costes de recursos humanos:

Rol	Salario Anual Bruto (€)	Seguridad Social Anual (30'15%) (€)	Total Horas Anual	Coste/ Hora (€)
Analista	42.500	12.814	1.728	32'01
Diseñador	48.000	14.472	1.728	36'15
Desarrollador	30.000	9.045	1.728	22'60
Ingeniero de pruebas	72.000	21.708	1.728	54'23
Responsable de documentación	60.000	18.090	1.728	45'19

Costes de personal durante el proyecto:

Rol	Horas	Coste/Hora(€)	Total(€)
Analista	180	32'01	5761,8
Diseñador	120	36'15	4338
Desarrollador	780	22'60	17628
Ingeniero de pruebas	40	54'23	2169,2
Responsable de documentación	150	45'19	6778,5
TOTAL	1270		36675,5

Evidentemente, para llevar a cabo el proyecto hay más gastos aparte del que conlleva la mano de obra, e incluye gastos de hardware, software, viajes y gastos fungibles (consumidos por el uso). La clasificación de estos gastos es la siguiente:

- Gastos hardware: se ha hecho uso de un ordenador portátil, ya que se puede desarrollar el proyecto casi completo (código, certificados, documentación, etc.) y ha sido útil para las tutorías.
- Material de oficina: papel, bolígrafos, encuadernaciones, etc.
- Material informático: cartuchos de tinta, CD, DVD, etc.
- Electricidad: energía necesaria para poder hacer funcionar el ordenador portátil.
- Viajes: durante el desarrollo del proyecto ha sido necesario realizar varias reuniones con el tutor.

Gastos de hardware, cálculo de amortización:

Equipo	Coste sin IVA (21%) (€)	Vida útil estimada en meses	Tiempo de uso en proyecto	Coste imputable al proyecto (€)
Ordenador Portátil Dell Latitude E5410. Intel Core i5 M520 2.4Ghz. 64 bits. 4Gb RAM	500	36	20	277

Gastos de software, cálculo de amortización:

Software	Coste de licencia sin IVA (21%) (€)	Duración de la licencia en meses	Tiempo de uso en proyecto	Coste imputable al proyecto (€)
Microsoft Office 2010 Home	139	36	20	76

Se excluye la licencia del sistema operativo Windows porque esta viene incluida en el precio del portátil.

En la siguiente tabla se muestra el coste de cada uno de los gastos clasificados anteriormente.

Descripción	Coste(€)
Gastos de hardware	277
Material de oficina	10
Material informático	10
Gastos de Software	76
Transporte	100

El importe total de los recursos materiales empleados asciende a un total de 473 €.

2.5.4 Resumen del presupuesto

La siguiente tabla muestra un resumen de todos los costes involucrados en el proyecto, así como la suma total de los mismos. Se establece un margen de riesgo del 15% y un beneficio económico del 20%. A continuación se muestra un resumen de los gastos desglosados en los apartados anteriores, añadiendo el IVA correspondiente al año 2012 (18%):

Recurso	Coste total(€)
Personal	36.675,5
Gastos	473
Total gastos	**37.148,5**
Prima por riesgo (15%)	5.772,3
Total incluido riesgo	**42.920,8**
Beneficio (20%)	8.584,16
Base imponible	**51.504,96**
I.V.A. (21%)	10.816,04
Total con I.V.A.	**62.321,00 €**

El presupuesto total del proyecto es de 62.321,00 €.

Leganés, 24 de Octubre de 2012
El ingeniero técnico proyectista: Fdo. Raúl David Martínez Calmaestra

3. ESTADO DE LA CUESTIÓN

3.1 Estado de la cuestión

3.1.1 Introducción

En este capítulo, introduciremos los conceptos de Servicio de Confianza, SASET (Servicios de Acreditación y Sellado Espacio Temporal). Explicaremos qué tipo de servicio es, en qué consiste y requisitos a cumplir.

Por otro lado, nos centraremos en un subconjunto de los SASET que se conoce como SST (Servicios de Sellado Espacio Temporal). El presente Proyecto de Fin de Carrera extiende las funcionalidades implementadas en otro proyecto anterior ([1]) que cubre dichos servicios SST. Dicho proyecto hacía un análisis de los posibles protocolos que implementan SST y finalmente concluía en implementar el software para uno de esos protocolos, que se denomina protocolo González Tablas, descrito en [2]. Nos centraremos en explicar en qué consiste dicho Protocolo y cual es la arquitectura seguida para implementarlo.

3.1.2 Estado de la cuestión

Este proyecto se enmarca dentro del ámbito de los Servicios de Confianza (SCZ) que a su vez pertenecen al área de la Seguridad de la Información.

Los Servicios de Confianza son provistos por unas entidades llamadas Terceros de Confianza (TTP, Trusted Third Parties). La definición según [13] es *"una organización o uno de sus agentes que provee de uno o más servicios de seguridad, y en la que otras entidades confían con respecto a actividades relacionadas con dichos servicios de seguridad"*.

Estas entidades aportan un valor añadido actuando como entidades intermedias en una relación entre dos partes. Una parte provee de una determinada información, y el "Tercero de Confianza" da fe de que dicha información es cierta ante cualquier otra parte, contando con el requisito de que la parte que provea dicha información y cualesquiera partes que hagan uso de ella, confían en dicho Tercero de Confianza. El Tercero de confianza aporta confidencialidad, integridad y disponibilidad de la información.

Esa información en el ámbito que nos ocupa es información espacio-temporal, y consiste en proporcionar la ubicación geográfica de una entidad que llamaremos Sujeto (S). El TTP proporciona una serie de evidencias probatorias de dicha información de ubicación espacio-temporal es exacta gracias a esa confianza dada por el Sujeto, y que en el ámbito que nos ocupa este proyecto se llaman Evidencias Espacio Temporales (EET).

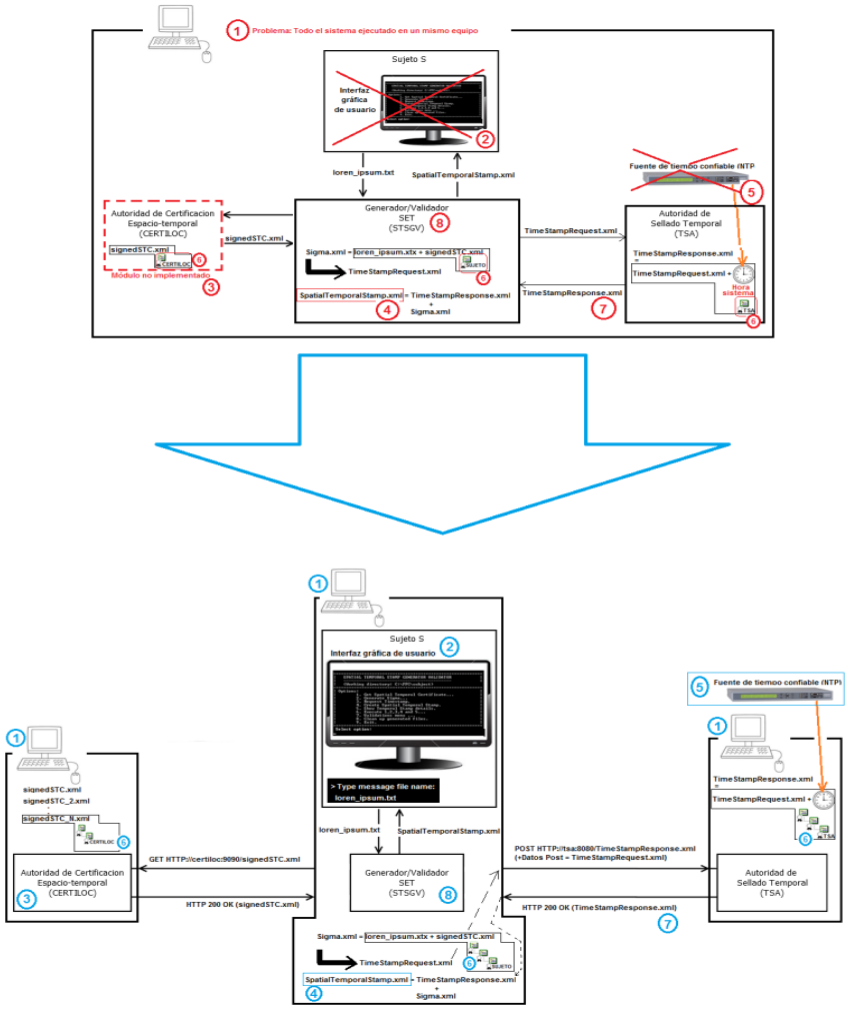
Como se explica en [1] y a continuación resumimos, entre los servicios de Confianza existen una serie de tipos como:

- *Servicios de Autenticación ("proceso de verificar la identidad del remitente de una comunicación")*
- *Servicios de Acreditación (demostración de la veracidad de una afirmación frente a otros, típicamente como los servicios de clave pública)*
- *Servicios de Notarización(servicios de No Repudio en los que un TTP da fe de la exactitud y veracidad de unos datos de una comunicación, y entre cuyos ejemplos se encuentran los Servicios de Sellado Temporal)*

Un **Servicio de Acreditación Espacio Temporal** es aquel proporcionado por una o varias entidades TTP en el que están involucradas como partes una entidad Sujeto que está relacionado con un determinado lugar (información espacial) y en un determinado momento (información temporal), y otra parte verificadora cuyo objeto es analizar dicha información espacio-temporal para verificar las condiciones de integridad, confidencialidad y no repudio de ésta. Ambas partes, Sujeto y Verificador, confían en el Tercero de Confianza.

4. ANÁLISIS DEL PROBLEMA Y ALTERNATIVAS DE SOLUCIÓN

El presente proyecto se basa en una implementación del protocolo ya realizada en [1]. Esta implementación, sin embargo, adolece de una serie de carencias que ya hemos citado grosso modo en la sección 1.2 Descripción del problema del presente proyecto. Este capítulo tiene como objeto describir más en detalle estas carencias y, como parte del análisis, proponer la manera de afrontar cada uno de esos problemas. El capítulo está organizado de la siguiente manera:



4.1 Descripción del proceso original, y principales problemas encontrados: En esta sección expondremos una ilustración del proceso en el que nos basamos, especificados en [1]. En rojo aparecerán numeradas las principales carencias funcionales y técnicas identificadas, que son 8 en total. Tambien describiremos los pasos que sigue el software implementado y por último proporcionamos una breve descripción de cada uno de los problemas encontrados. Algunos de los puntos aquí encontrados aparecen en el capítulo Líneas Futuras de la memoria del proyecto [1].

4.2 Propuestas de solución para los problemas descritos: En esta sección expondremos una ilustración que muestra la nueva arquitectura cliente-servidor implementada, y aparecen numerados en azul las soluciones aportadas a cada uno de los 8 problemas encontrados en la sección anterior. Tambien describiremos los pasos a seguir por el software implementado y por último proporcionamos una breve descripción de cada una de las soluciones aportadas para cada uno de los 8 problemas que se encontraron.

Por otro lado, las soluciones citadas en la sección 4.2 están explicadas y detalladas en las sucesivas secciones 5.1 a 5.8, del capítulo *5 Análisis y Diseño*.

Por último, en la sección 4.3 enumeramos otros problemas de carácter más técnico, y no por ello menos importantes, que se han encontrado en la implementación realizada en [1]. Se trata de una serie de carencias técnicas junto con la solución propuesta.

4.1 Descripción del proceso original, y principales problemas encontrados

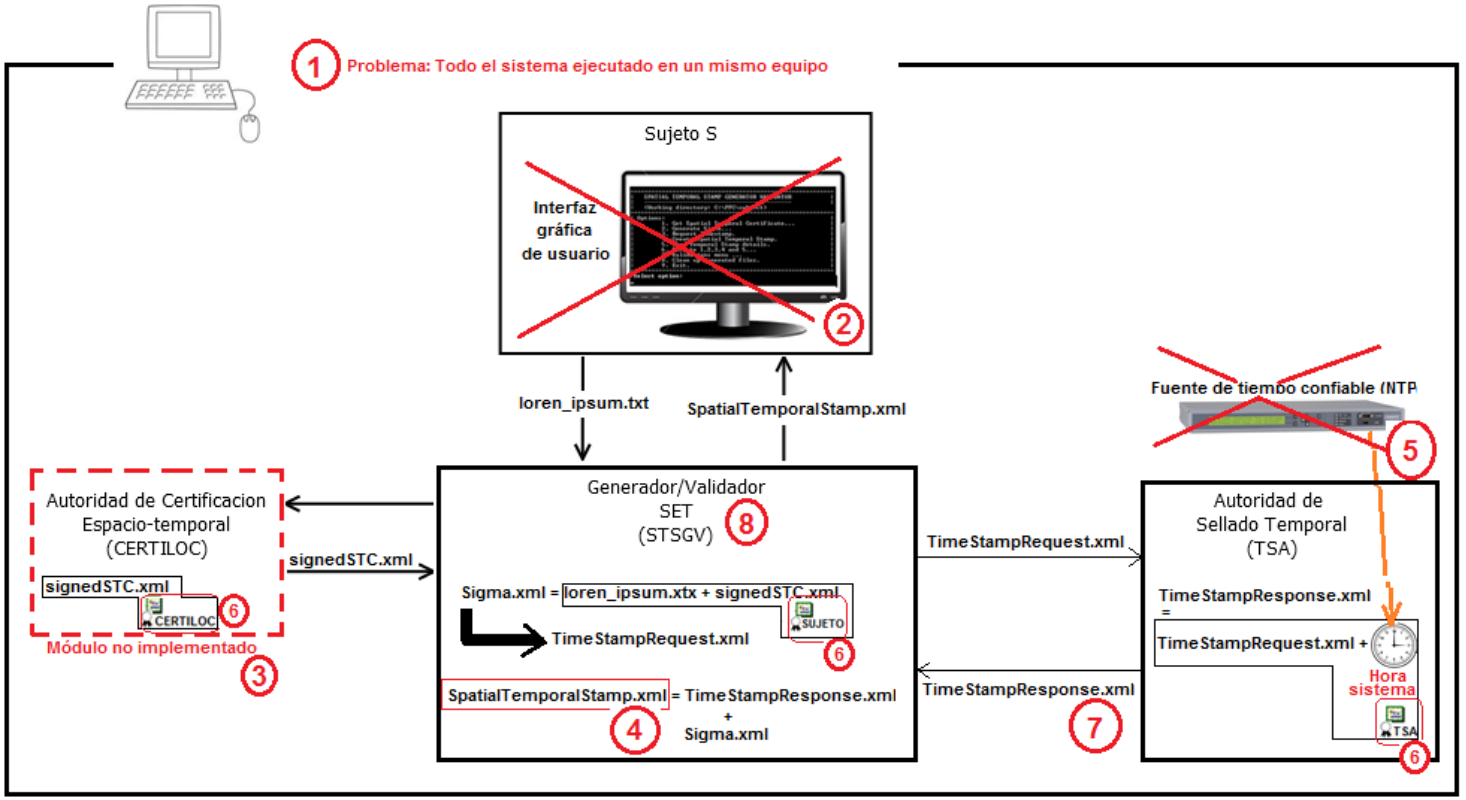
Descripción de los pasos que sigue el proceso:

I) El módulo Generador/Validador toma los archivos Mensaje (loren_ipsum.txt) y Certificado Espacio Temporal (signedSTC.xml). Con ellos compone un archivo Sigma.xml firmado por el Sujeto

II) El módulo Generador/Validador: Con Sigma.xml y más información (prefijada por código de la aplicación) compone el archivo de petición de tiempo(TimeStampRequest.xml) a la TSA (SST, Servicio de Sellado de Tiempo)

III) Módulo TSA: El archivo de petición de sello de tiempo TimeStampRequest.xml es tomado por el módulo TSA, se añade la información de tiempo y se compone el documento de respuesta de Sello de Tiempo (TimeStampResponse.xml), que va firmado por la TSA.

IV) El módulo Generador/Validador toma el documento de respuesta de la TSA y lo combina con el documento Sigma generado en el paso I, para generar el Sello Espacio Temporal final (SET).



1. **Problema**: **Ejecución monopuesto**. Las entidades principales (Sujeto,TSA,CERTILOC) se ejecutan en el mismo equipo, asumiendo que todos los xml generados se generan en el mismo directorio físico del mismo ordenador.	**5.** **Problema**: **Falta de confianza en la fuente de información temporal**. La TSA genera la información de tiempo a devolver tomando la fecha del sistema.
2. **Problema**: **Ausencia de una interfaz de usuario amigable**. Para ejecutar cada uno de los pasos del protocolo se debe ejecutar una sentencia de consola distinta para cada uno.	**6.** **Problema**: **Carencia de una estructura suficiente de entidades de certificación**. Las entidades CERTILOC, TSA y Sujeto son de certificado único autofirmado. Además los certificados de clave pública son de la versión V1 de X.509. Debería haber al menos alguna Autoridad de Certificación involucrada.
3. **Problema**: **Falta de implementación del módulo CERTILOC**. La implementación de partida ([1]) se basa solo en un archivo Certificado Espacio Temporal que siempre es el mismo.	**7.** **Problema: Incorrecto tratamiento del Nonce**. El nonce es un número que se genera aleatoriamente para cada transacción de petición a la TSA. La TSA debe retornar el mismo nonce que se le envió en la petición. Actualmente esto no lo hace correctamente
4. **Problema**: **Falta de semántica de los documentos generados**. En algunos de los nodos de los xml donde se hace referencia a documentos externos, no hay información sobre la ruta de dichos documentos ni se incluye un resumen digital de sus contenidos y falta el tsp:References en el SET.	**8.** **Problema**: **Ausencia de validación de los certificados**. Estos certificados van incluídos en las firmas XML Signature.

4.2 Propuestas de solución para los problemas descritos

I) Generador/Validador: Se solicita al usuario que teclee el nombre del fichero de mensaje, y el nombre del fichero de CET a solicitar a CERTILOC

II) Generador/Validador hace una petición HTTP al servidor CERTILOC, que devolverá un mensaje HTTP con el contenido del CET solicitado.

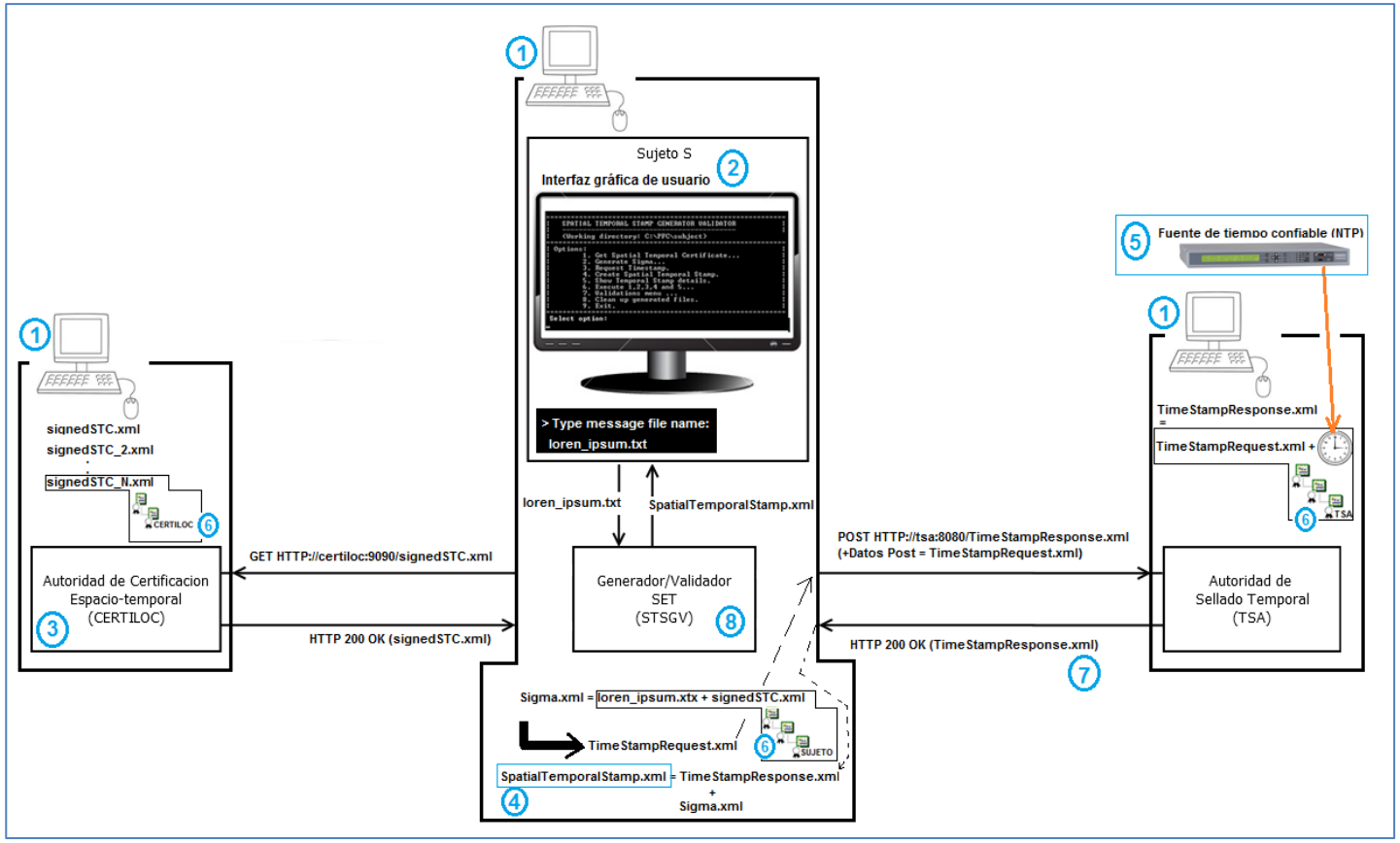
III) Generador/Validador toma los archivos Mensaje (loren_ipsum.txt) y CET (signedSTC.xml). Con ellos crea un archivo Sigma.xml firmado por el Sujeto

IV) Generador/Validador: Con Sigma.xml y más información obtenida de archivo de configuración compone el archivo de petición de tiempo(TimeStampRequest.xml)

V) Generador/Validador hace una petición HTTP al servidor TSA. Esta petición es de tipo POST pues incluye el contenido del archivo de petición, y el archivo a solicitar es el archivo de respuesta TimeStampResponse.xml

VI) TSA: El archivo de petición de sello de tiempo es procesado, se añade la información de tiempo y se compone el xml de respuesta de Sello de Tiempo firmado por la TSA.

VII) Generador/Validador toma el documento de respuesta de la TSA y lo combina con el documento Sigma del paso I, para generar el Sello Espacio Temporal.



1. Solución a "Ejecución monopuesto". La solución propuesta es implementar un cliente-servidor en el que el Generador/Validador sea cliente de los servidores CERTILOC y TSA por un medio de HTTP	**5. Solución a "Falta de confianza en la fuente de información temporal"**. Se modificará el módulo de la TSA para que haga una petición mediante protocolo NTP a algun NTP público.
2. Solución a "Ausencia de una interfaz de usuario amigable". La solución propuesta consiste en hacer un sistema de menus y submenús en consola en el módulo Sujeto.	**6. Solución a "Carencia de una estructura suficiente de entidades de certificación"**. Se creará una estructura con una Autoridad de Certificación raíz y otras AC's intermedias.
3. Solución a "Falta de implementación del módulo CERTILOC". Se implementará este módulo como un servidor Http cuya única misión será devolver el archivo CET solicitado.	**7. Solución a "Incorrecto tratamiento del Nonce"**. La TSA procesará la petición recibida para devolver el mismo Nonce en su respuesta.
4. Solución a "Falta de semántica de los documentos generados". Se modificarán los documentos xml para que incluyan un atributo URI indicando la ubicación física del fichero referenciado y un resumen digital del mismo. Se incluirá un nodo tsp:References en el Sello Espacio Temporal, como especifica [5.4]	**8. Solución a "Ausencia de validación de los certificados"**. Se validará que los subnodos X509Data de las firmas XMLSignature contengan certificados de clave pública válidos.

4.3 Enumeración de otros problemas y propuestas de solución

- **4.3.1 Registro de números de serie:** La TSA no guarda un registro de los *serial number* entregados a las entidades Sujeto, siempre envía el mismo, ni siquiera generado aleatoriamente.
  Solución: Guardar un registro de los serial number entregados con éxito empezando por 1 e ir incrementándolo sucesivamente

- **4.3.2 Organización de código:** Los archivos que componen el software se encuentran sin una organización por carpetas y no están preparados para ser editados con una herramienta Entorno Integrado de Desarrollo (IDE)
  Solución: Crear un espacio de trabajo con alguna de las herramientas IDE existentes en el mercado, como NetBeans, Eclipse o JDeveloper. Se propone usar Eclipse dado que es gratuito, potente, extensible y por su popularidad. Se propone crear un workspace de Eclipse y dentro del mismo separar nuestro código en varios proyectos java permitiéndonos modularizar el código.

- **4.3.3 Organización de paquetes:** El código no está organizado en paquetes (packages) de java
  Solución: Organizar el código dentro de un paquete principal subdividido en varios paquetes correspondientes a cada módulo.

- **4.3.4 Carencias de modularidad:** La clase FirmaDigital no es suficientemente genérica, ejecuta distinto código según se trate de una invocación para la firma de la tsa o la firma del sujeto.
  Solución: Realizar un componente de firma nuevo que sea lo suficientemente genérico, es decir que permita realizar cualquier firma de cualesquiera nodos del documento xml que sea.

- **4.3.5 Encapsulamientos incorrectos:** Por ejemplo, el constructor de la clase RequestManager recibe como parámetros una serie de clases que son agregaciones unas de otras:

```
public RequestManager()
	{
		tsr = new TimeStampRequest("TimeStamp
		spi =  new SignaturePolicyIdentifier(
		sigpoli = new SigPolicyId();
		sigPolId = new SignaturePolicyId(sigp
		identifier =  new IdentifierType("htt
		sph = new DigestAlgAndValueType();
	}
```

  Solución: Intentar separar mejor las clases que están contenidas unas dentro de otras en función de una jerarquía comparable a los xml resultantes.

- **4.3.6 Constructores sobrantes:** Diversos constructores de clase que son innecesarios al ser constructores por defecto
  Solución: Estos constructores se eliminan.

- **4.3.7 Innecesario uso de herencia de clases:** como en el ejemplo de *"public class DigestAlgValue extends DigestAlgValueType"* porque una clase como DigestAlgValue es ya un tipo en sí misma. En el anterior proyecto se tomó DigestAlgValueType por concordancia con el esquema xml de xades. Esta herencia se torna inútil puesto que en ningún momento se hace upcasting de la subclase DigestAlgValue ni se hace uso alguno de la superclase DigestAlgValueType.
  Solución: Eliminar las herencias de clases innecesarias.

- **4.3.8 Depencencia de demasiadas librerías externas:** Por ejemplo, *jars* de terceros: hay redundancia, algunas incluso están en desuso, y de algunas no está documentado cual es su origen ni versión de software.
  Solución: Identificar las dependencias de librerías necesarias por funcionalidades (manejo de xml, seguridad/cifrado, etc) y documentar la fuente de cada una y la versión utilizada.

- **4.3.9 Ausencia de una estrategia de control de excepciones:** limitándose la misma a poner sentencias Try-catch en los sitios en los que el compilador obliga.
  Solución: Para las excepciones no previstas por el programa, establecer un convenio de manejo de excepciones, aunque sea poco sofisticado, como por ejemplo emplear la cláusula throws para que cada método relance las excepciones al método invocante y poner un try-catch padre que recoja todas las excepciones no manejadas. Excluir de este tratamiento aquellas excepciones previstas como ficheros no encontrados, o errores de introducción de clave de acceso a almacén de claves, etc porque dichas excepciones pueden ser manejadas con intervención del usuario

- **4.3.10 Tratamiento de XML mejorable:** El tratamiento de xml en memoria mediante DOM es mejorable en cuanto a la librería usada. Se utiliza la conocida librería Xerces de Apache, pero en los últimos años se ha popularizado el uso de nuevas librerías para el lenguaje java que permiten el manejo de xml de manera mucho más cómoda, como XOM, DOM4J o JDOM
  Solución: Se ha escogido la librería JDOM, parece que es más popular y por lo tanto más fácil de encontrar tutoriales y ayuda en internet. Por otro lado, al contrario que XOM y DOM4J JDOM es compatible con el sistema operativo Android, lo que hace este software más extensible en el uso para aplicaciones móviles.

- **4.3.11 Falta de configurabilidad:** En el código están hardcodeados (puestos literalmente) muchas cadenas de texto variable, de manera que algunas propiedades de la aplicación susceptibles de ser cambiadas obligan a recompilar el código en dicho caso.
  Particularmente crítico es eliminar el "hardcodeo" del año 2008 en el código que se emplea para extraer el gml:timePosition del CET
  Solución: Hacer uso de un archivo de texto externo al código cuyo contenido sea para cada línea un par de clave-valor, de manera que se puedan cambiar los valores sin necesidad de recompilar el código. Todos estos valores se explican en el apéndice C Mantenimiento

- **4.3.12 Inconsistencia idiomática:** Parte del código está en idioma inglés y parte en castellano
  Solución: Homogeneizar el nombrado de clases, propiedades, comentarios en un único idioma

- **4.3.13 Nombres de parámetros y variables poco descriptivos:** ("file1", "Certificate_1" "i", etc)
  Solución: Poner nombres autoexplicativos como "sMessageFile" o "Certificate_of_Subject".

- **4.3.14 Redundancias en documento:** Redundancia de los elementos X509Data y KeyValue hijos del nodo KeyInfo en las firmas ds:signature generadas
  Solución: Según el esquema de XML Signature, el KeyInfo (que contiene la clave pública del firmante) es un "choice" entre varios posibles subnodos. Proponemos quitar el subnodo KeyValue a favor del subnodo X509Data, más descriptivo al contener un certificado X509.

5. ANÁLISIS Y DISEÑO

5.1 Problema 1: Ejecución en monopuesto

Como mencionamos en la sección 1.2, el software presentado en [1] tiene el problema de que está hecho para ser ejecutado en una misma máquina, asumiendo que todos los documentos xml involucrados en el proceso de generación de un Sello Espacio Temporal se encuentran físicamente en el mismo directorio de la misma.

En un entorno real tendríamos una arquitectura cliente-servidor en la que la entidad Sujeto y el módulo Sellador-Validador actuarían como clientes, y por otro lado tendríamos como servidores CERTILOC (que provee del servicio de Acreditación Espacio Temporal) y una Autoridad de Sellado Temporal (que provee del servicio de sellado Temporal).

Es por ello que nuestra propuesta es modificar el software presentado en [1] para que tenga una arquitectura cliente-servidor y que la comunicación entre dichos clientes y servidores se realice por un medio de transmisión en red.

Se estudiaron diversas opciones para realizar la transmisión, como por ejemplo transmisión mediante mensajes SOAP o transmisión mediante ServerSockets gestionados por la interfaz Runnable de java , pero finalmente encontramos como solución más óptima el uso del protocolo HTTP como medio de transmisión.

Según [27], *Hypertext Transfer Protocol o HTTP (en español protocolo de transferencia de hipertexto) es el protocolo usado en cada transacción de la World Wide Web.* (…) *Es un protocolo orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. Al cliente que efectúa la petición (un navegador web o un spider) se lo conoce como "user agent" (agente del usuario). A la información transmitida se la llama recurso y se la identifica mediante un localizador uniforme de recursos (URL). Los recursos pueden ser archivos, el resultado de la ejecución de un programa, una consulta a una base de datos, la traducción automática de un documento, etc.*

Según esta definición, en el entorno que nos ocupa podemos hacer la siguiente identificación:
User agent: En nuestro sistema se trata del conjunto Sujeto-Sellador/Validador. Actúa de cliente HTTP
Recurso: En nuestro sistema se trata de documentos xml que contienen Certificado Espacio Temporal, Petición de Sello de Tiempo (Timestamp Request) y Respuesta de Sello de Tiempo (Timestamp Response)

Por otro lado, también según [27] *HTTP es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores. El desarrollo de aplicaciones web necesita frecuentemente mantener estado*. En nuestro sistema el estado es un concepto que abarca "en qué paso del protocolo de Sellado Espacio Temporal nos encontramos". Mientras que en el entorno hipertextual de la world wide web este estado nos llevaría al concepto de sesión y el uso de "cookies", en nuestro sistema el estado nos viene dado por los diversos documentos que se van creando en el Sujeto, CERTILOC y la TSA. Dicho de otra manera, se puede inferir en qué parte del proceso nos encontramos por la presencia o ausencia de los documentos xml, que además están relacionados entre sí criptográficamente mediante firmas XML Signature y mecanismos de resumen digital.
Según [27], *HTTP define 8 métodos (algunas veces referido como "verbos") que indica la acción que desea que se efectúe sobre el recurso identificado.* Entre dichos métodos se encuentran HEAD, GET, POST, PUT…

En cuanto al formato de petición de recursos a emplear en nuestro sistema, proponemos el uso de los dos siguientes métodos de petición HTTP: GET y POST. GET nos permite solicitar un determinado recurso. Uno de estos ejemplos en nuestro contexto es la petición de una Acreditación Espacio Temporal por parte del Sujeto a CERTILOC, en la que el recurso a solicitar es dicha Acreditación Espacio-Temporal.

El método HTTP POST nos permite solicitar un determinado recurso y a la vez enviar unos datos junto con la petición, que permitirán al servidor conocer más información sobre la misma. En nuestro contexto utilizaremos este método para la petición de Sello de Tiempo por parte del Sujeto a la TSA, porque junto con la petición necesitamos enviarle a la TSA el documento XML que conocemos como Documento de Petición de Sello de Tiempo (TimeStamp Request).

De esta manera, la entidad Sujeto funciona como un cliente HTTP que realizará peticiones GET o POST a los servidores http, que son dos: uno para la TSA y otro que hace de simulador del software de CERTILOC.

El esquema es el siguiente:

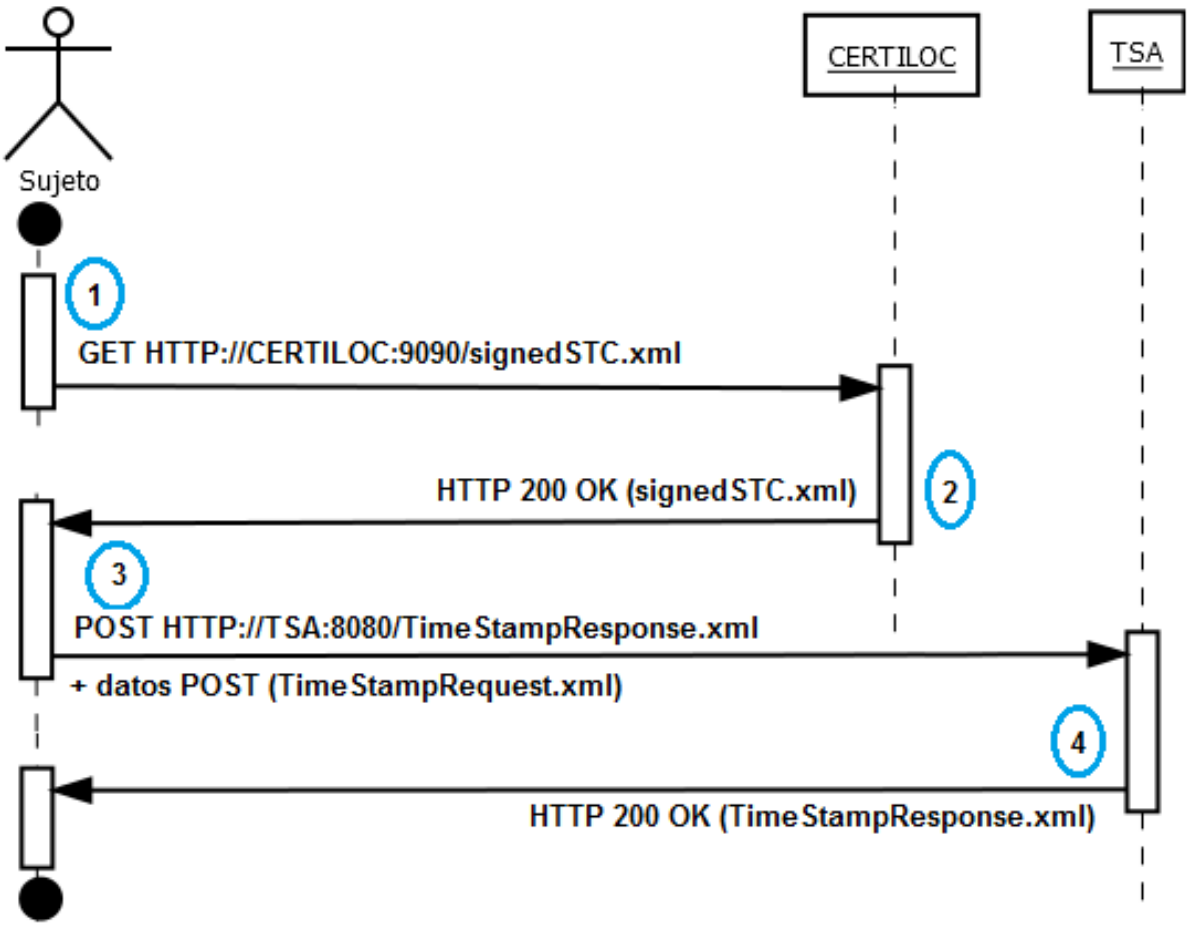


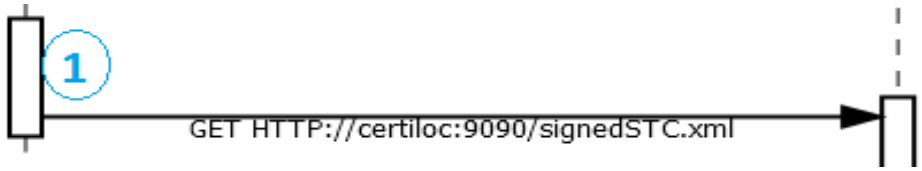
Figura 5.1: Diagrama de secuencia de transmisión

A continuación detallamos cada uno de los intercambios de información entre las entidades. El funcionamiento de dichos intercambios puede demostrarse en la sección 7.6 Pruebas de petición y respuesta HTTP:

1) **Petición de Acreditación Espacio Temporal**:

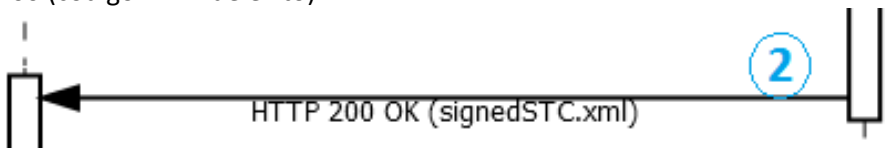
En la consola del Sujeto, se pide al usuario que especifique el nombre del archivo mensaje, y a continuación el nombre del xml que contiene el certificado de CERTILOC que desea utilizar. Entonces se realiza una petición HTTP al servidor de CERTILOC de tipo GET, para solicitar el archivo xml . La petición tiene el formato:

GET HTTP://[NOMBRE SERVIDOR CERTILOC]:[NºPUERTO]/[RUTA/AL/ARCHIVO]/[NOMBREARCHIVO]



Tanto el nombre del servidor, como el número de puerto de escucha, como el directorio de funcionamiento del servidor son parametrizables como se explica en la sección del Manual de Usuario Parámetros del software.

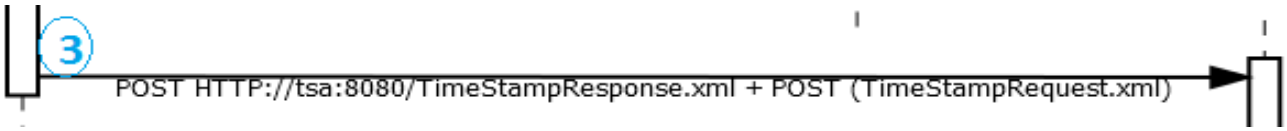
2) El servidor de CERTILOC servirá el xml del Certificado Espacio Temporal en la respuesta HTTP con un código 200 (código HTTP de éxito).



3) **Petición de Sello Espacio Temporal**:

El sujeto toma el Certificado Espacio Temporal, el Mensaje y calcula su resumen. Este resumen lo mete en un xml de petición (TimeStampRequest.xml) y realiza una petición de tiempo a la TSA, mediante una petición HTTP de tipo POST, solicitando el archivo TimeStampResponse.xml, y enviando en el cuerpo (body) de dicha petición como datos POST el contenido del archivo TimeStampRequest.xml. El formato de dicha petición será el siguiente:

POST HTTP://[NOMBRE SERVIDOR TSA]:[NºPUERTO]/[RUTA/AL/ARCHIVO]/TimeStampResponse.xml



4) Si la petición no tiene ningún problema, la TSA devolverá un código 200 de HTTP junto con la respuesta Timestamp Response en el cuerpo de dicha respuesta.

5) Se realizarán los pasos restantes del Protocolo de Sellado Espacio Temporal por parte del Sujeto.

5.2 Problema 2: Ausencia de una interfaz de usuario amigable

En el sistema presentado en [1] el esquema de funcionamiento era el siguiente: el punto de entrada de la aplicación es una clase java principal y ejecutable llamada STSGeneratorValidator. Para ejecutarla hay que utilizar el comando java + STSGeneratorValidator + un carácter (f,r,s,v) que representa el paso del protocolo a ejecutar. Por otro lado, para la TSA se utiliza una clase ejecutable distinta llamada TSAManager, lo que añade un comando más a ejecutar. Esta forma de funcionamiento es poco intuitiva para el usuario y proporciona poca información en pantalla.

Por este motivo se ha estudiado cómo podemos proporcionar a la aplicación una interfaz sencilla y manejable que nos permita conocer en todo momento las diversas opciones disponibles para ejecutar y que nos proporcione la suficiente información en pantalla. Hemos optado por implementar un sistema de menus en consola de comandos, que se puede ejecutar sin necesidad de ningún software especial más que un entorno de ejecución java y que es independiente de la plataforma de ejecución.

El funcionamiento de la interfaz es el siguiente si lo desglosamos por entidades del protocolo de Sellado:

- Sujeto-Sellador/Validador: Sistema de menus y submenús que contiene diversas opciones, que, entre otras opciones, corresponden con pasos del protocolo de sellado espacio temporal.
- Autoridad de Sellado Temporal (TSA): no habrá interfaz, al ejecutarse se pone en funcionamiento esperando peticiones entrantes, y mostrará información en pantalla sobre cada una de ellas.
- Autoridad de Acreditación Espacio-Temporal (Simulador de CERTILOC): no habrá interfaz, al ejecutarse se pone en funcionamiento esperando peticiones entrantes, y mostrará información en pantalla sobre cada una de ellas.

FUNCIONAMIENTO ANTERIOR	NUEVO FUNCIONAMIENTO PROPUESTO
1) Comando “Java STSGeneratorValidator f “ Se crean documentos preliminares contenedores de la entidad Sigma, llamados dataToBeSigned.xml y SpatialTemporalStamp.xml	Preparar los archivos de configuración STSS.properties en los tres entornos de ejecución (Sujeto, TSA, CERTILOC) para que estén establecidos los puertos de escucha TCP correctos de las distintas máquinas y demás elementos.
2) Comando “Java STSGeneratorValidator r” Se genera el TimeStampRequest.xml a partir de los documentos generados en el paso anterior.	1) En la máquina donde está instalada la TSA: Comando “Java TSA.jar”: La TSA quedará a la escucha de nuevas peticiones de Sello de Tiempo entrantes
3) Comando “Java TSAManager” La TSA genera el TimeStampResponse.xml a partir del documento generado en el paso anterior	2) En la máquina donde está instalado CERTILOC: Comando “Java CERTILOCSIM.jar”: CERTILOC quedará a la escucha de nuevas peticiones de Certificados Espacio Temporales.
4) Comando “Java STSGeneratorValidator s” El Sujeto crea el Sello Espacio Temporal (SpatialTemporalStamp.xml)	3) En la máquina donde está instalado el Sujeto: Comando “Java STSS.jar”: Se mostrará un menú con las diferentes opciones a elegir.
5) Comando “Java STSGeneratorValidator v” Se valida el Sello Espacio Temporal.	4) En la máquina donde está instalado el Sujeto: Elegir las opciones de menú necesarias (según si es usuario avanzado o normal). El programa solicitará la información necesaria como nombre del documento Mensaje a incluir y nombre del Certificado Espacio Temporal a solicitar y generará el documento Sello Espacio Temporal (SpatialTemporalStamp.xml)
	5) En la máquina donde está instalado el Sujeto: Elegir las opciones de menú necesarias (según si es usuario avanzado o normal) El programa validará que las firmas y resúmenes incluídos en el SET son correctos.

Una consideración que hemos hecho en el presente proyecto es que hemos separado dos tipos de usuarios de la interfaz Sujeto-Sellador/Validador: Por un lado tendremos un usuario "avanzado" que podrá ejecutar independientemente los distintos pasos que sigue el protocolo de Sellado Espacio Temporal, y por otro lado tendremos el usuario "normal", que tendrá la opción de ejecutar el proceso completo sin poder ejecutar cada uno de los pasos por separado.

Por este motivo hemos realizado dos posibles escenarios de casos de uso para nuestra aplicación. Ambos escenarios tienen en común la parte correspondiente a la Autoridad de Sellado Temporal y a la parte de la Autoridad de Certificación Espacio Temporal.

Diagrama de casos de uso para el escenario de usuario "normal":

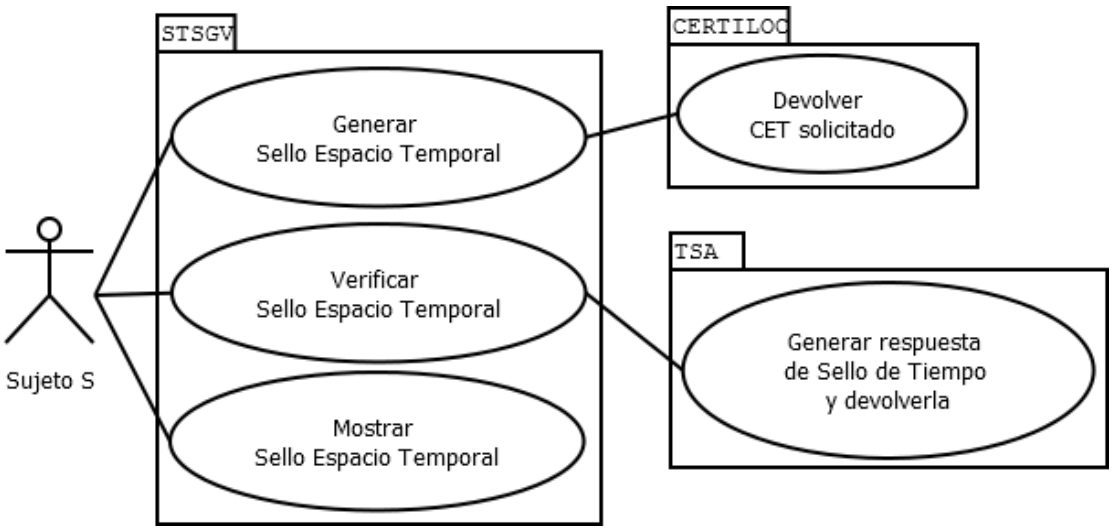


Figura 5.2.1 Diagrama de Casos de uso para usuario normal

El conjunto de casos de uso propuesto para el escenario de usuario "normal" tiene traslación directa en el sistema de menus que proponemos para la interfaz gráfica de usuario. Estructura propuesta para el menu de usuario normal:

OPCIÓN DE MENU	Propósito
1) Generate Spatial Temporal Stamp (Crear Sello Espacio Temporal)	Realizar una petición de una acreditación espacio temporal (la que teclee el usuario) al simulador de certiloc. Generar el documento Sigma a partir de los documentos Mensaje (a indicar por el usuario) y CET. Genera un documento de petición de Sello de Tiempo (TimeStamp Request) y envía una petición a la TSA que contiene dicho documento, recibiendo un documento de Respuesta de Sello de Tiempo (TimeStamp Response) Genera el Sello Espacio Temporal final a partir del Sigma generado en el paso 2 y de la respuesta de Sello de Tiempo.
2) Verify STS	Realiza todas las validaciones de firmas, certificados y resúmenes sobre el Sello Espacio Temporal y ficheros relacionados.
3) Show Spatial Temporal Stamp details (Mostrar detalles del Sello Espacio Temporal)	Muestra la información más relevante del Sello Espacio Temporal
4) Exit (Salir)	Salir del programa

En el proceso de firma por parte del Sujeto de la entidad Sigma, al acceder al almacén de claves del Sujeto se necesita una passphrase de acceso. Tenemos una propiedad de configuración llamada SUBJECT.CONSOLEUI.ASKFORCERTPASSWORD que nos permite establecer si queremos que el usuario teclee la clave, o bien que no necesite hacerlo pues existen otra propiedad de configuración donde el valor de dicha clave aparece. Por motivos obvios de seguridad, se recomienda que cuando el escenario de ejecución de menú sea el de usuario normal, se solicite la clave al usuario, por tanto esta propiedad se recomienda que esté con el valor true (verdadero).

Diagrama de casos de uso para el escenario de usuario “avanzado”:

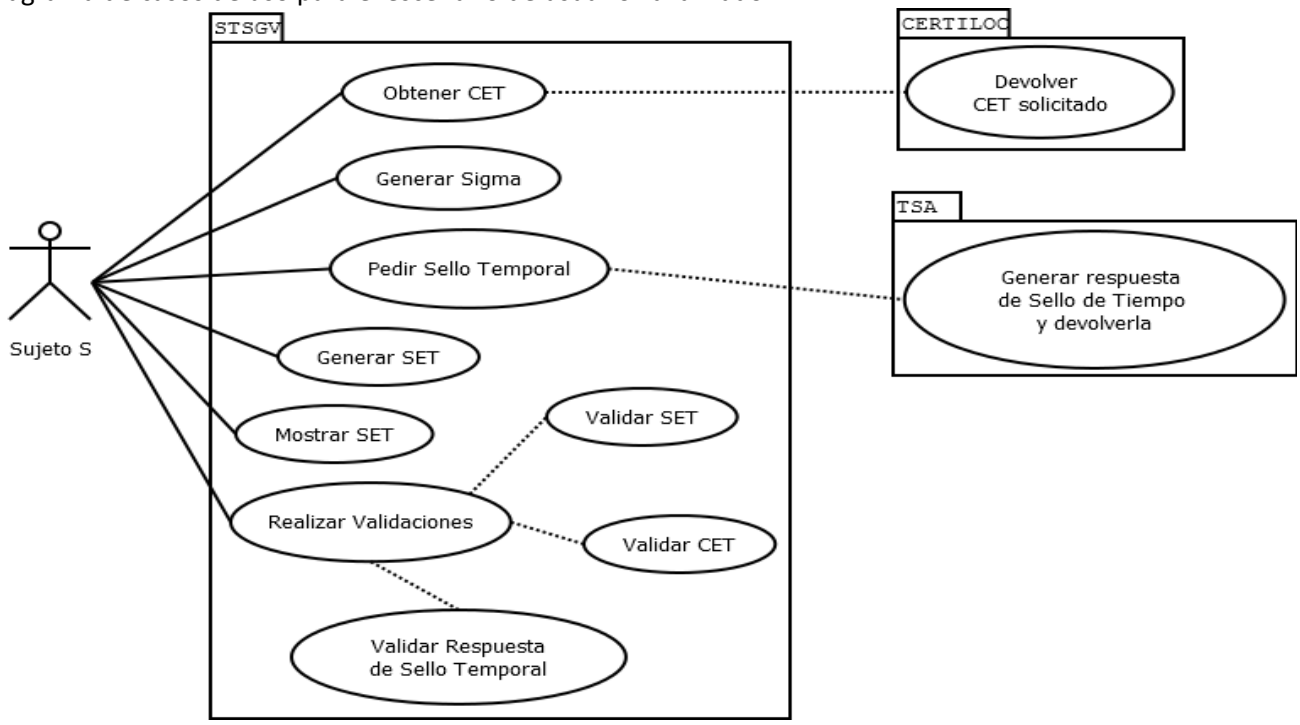


Figura 5.2.2 Diagrama de casos de uso para usuario avanzado

El conjunto de casos de uso propuesto para el escenario de usuario “avanzado” tiene traslación directa en el sistema de menus que proponemos para la interfaz gráfica de usuario. Estructura propuesta para el menu de usuario avanzado:

OPCIÓN DE MENU	SUB-OPCIÓN DE MENU	Propósito
1) Get Spatial Temporal Certificate (obtener certificado espacio temporal)		Realizar una petición de una acreditación espacio temporal al simulador de certiloc
2) Generate Sigma (generar Sigma)		Generar el documento Sigma a partir de los documentos Mensaje (a indicar por el usuario) y CET (a pedir por el usuario en el paso 1)
3) Request Timestamp (solicitar Sello de Tiempo)		Genera un documento de petición de Sello de Tiempo (TimeStamp Request) y envía una petición a la TSA que contiene dicho documento, recibiendo un documento de Respuesta de Sello de Tiempo (TimeStamp Response)
4) Create Spatial Temporal Stamp (Crear Sello Espacio		Genera el Sello Espacio Temporal final a partir del Sigma generado en el paso

Temporal)		2 y de la respuesta de Sello de Tiempo.
5) Show Spatial Temporal Stamp details (Mostrar detalles del Sello Espacio Temporal)		Muestra la información más relevante del Sello Espacio Temporal
6) Execute 1,2,3,4 and 5 (Ejecutar pasos 1 a 5)		Ejecuta los pasos 1 a 5 (con las interacciones con el usuario necesarias) y como resultado final genera el Sello Espacio Temporal
7) Validations menú (menú de validaciones		Mostrar el submenú de validaciones
	1) Validate Spatial Temporal Stamp (Validar Sello Espacio Temporal)	
	2) Validate Spatial Temporal Certificate (Validar Certificado Espacio Temporal)	
	3) Validate TimeStamp Response (Validar Respuesta de Sello de Tiempo)	
	4) Validate all (Validar todo)	
	5) Return to main menú (Volver al menú principal)	
8) Clean up generated files (Limpiar los archivos generados)		Borrar todos los archivos generados
9) Exit (Salir)		Salir del programa

5.3 Problema 3: Falta de implementación del módulo CERTILOC

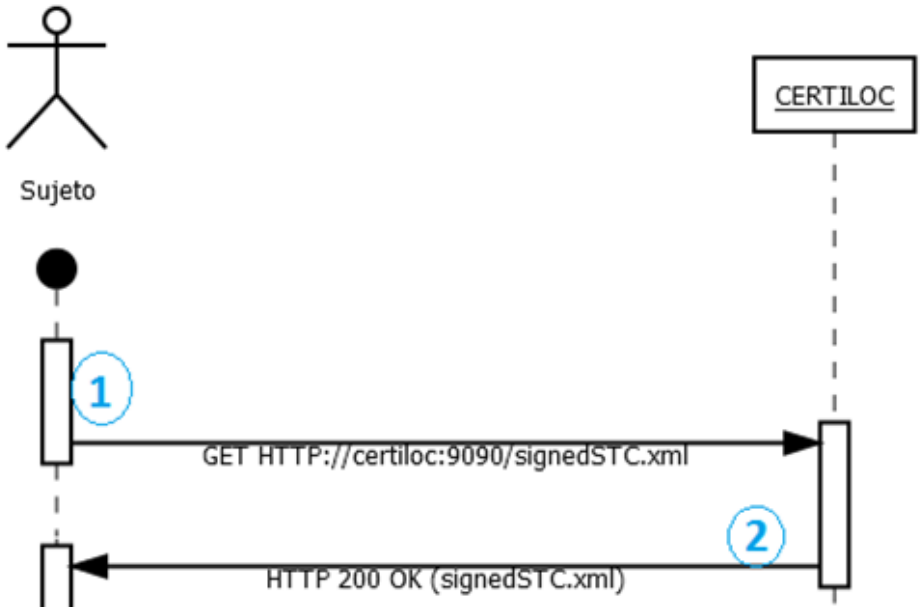
Este problema ya fue identificado como una línea futura a seguir en el proyecto de partida ([1]). En concreto lo que se especifica es lo siguiente, en el capítulo 6.2 Líneas Futuras del mismo:

Integración en el núcleo del demostrador de CERTILOC. Al haberse desarrollado todos los componentes para trabajar "en local" (en la misma máquina), no se ha tenido en cuenta la implementación de ningún mecanismo de comunicación con máquinas remotas. Debido a que el módulo de la TSA irá integrado en el servidor de CERTILOC, debe implementarse un mecanismo (via servlets o via web) con usuarios remotos.

Hemos descartado la posibilidad de integración total con la implementación existente del Proyecto CERTILOC, en su lugar hemos hecho una propuesta práctica que consiste en implementar un "Simulador de CERTILOC", que consiste en un servidor HTTP encargado de servir Certificados Espacio Temporales, con la doble posibilidad de que:

- bien dichos CET estarán ubicados físicamente en el servidor donde funcione dicho simulador
- o bien se generan en el momento. El cliente (Sujeto) tendrá la posibilidad de escoger entre las dos opciones en el proceso de generar un sello espacio temporal.

Se puede ver la relación con la sección 5.1 Problema 1: Ejecución en monopuesto, en la parte donde se describe la petición de Acreditación Espacio Temporal y que citamos a continuación:



Petición de Acreditación Espacio Temporal:
En la consola del Sujeto, se pide al usuario que especifique el nombre del archivo xml que contiene el certificado de CERTILOC que desea utilizar. Entonces se realiza una petición HTTP al servidor de CERTILOC de tipo GET, para solicitar el archivo xml . La petición tiene el formato:
GET HTTP://[NOMBRE SERVIDOR CERTILOC]:[NºPUERTO]/[RUTA/AL/ARCHIVO]/[NOMBREARCHIVO]
Tanto el nombre del servidor, como el número de puerto de escucha, como el directorio de funcionamiento del servidor son parametrizables como se explica en la sección del Manual de Usuario Parámetros del software.

En el caso especial de que el nombre de fichero CET pedido sea “signedSTC.xml”, el servidor CERTILOC está programado para emitir un CET nuevo con la fecha actual. En cuanto a la información espacial, al tratarse de un simulador, CERTILOC asignará una localización aleatoriamente escogida de un fichero que contiene varias localizaciones escogidas arbitrariamente llamado dummyLocations.txt, configurable con el parámetro CERTILOCSIM.FILES.DUMMYLOCATIONSFILE. El CET resultante de combinar la información espacial con la información temporal actualizada se firmará digitalmente en el momento con las credenciales de CERTILOC de manera que se pueda comprobar posteriormente la integridad criptográfica de la Acreditación Espacio Temporal.

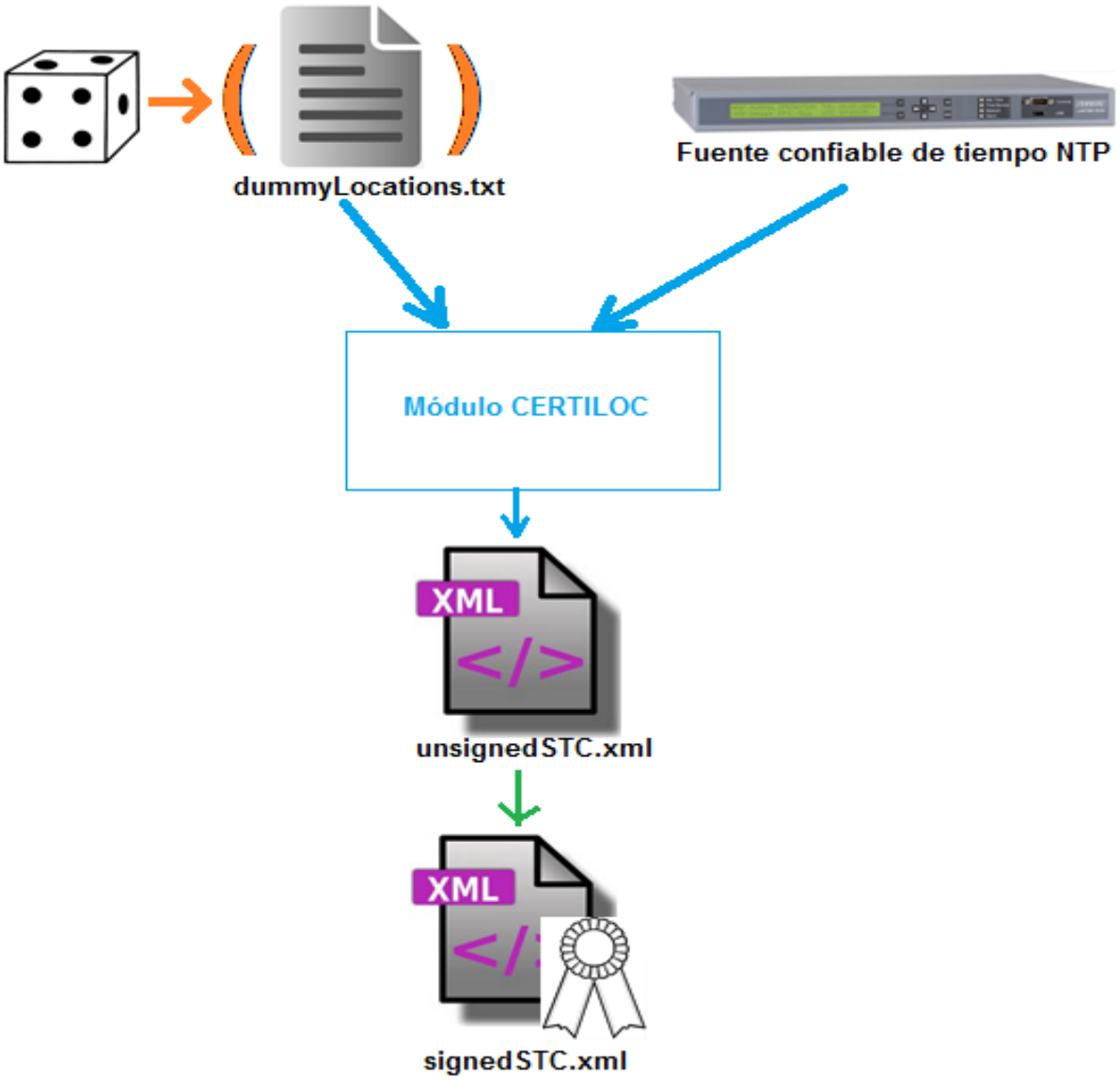


Figura 5.3 Esquema de creación de nuevo CET

Es necesario explicar en este punto que tanto la información temporal como la información espacial, serán incluídas en un documento llamado unsignedSTC.xml que nos sirve a modo de plantilla para incluir dicha información. Como con este documento no es suficiente, el mismo será firmado con la clave privada de CERTILOC y dicho documento firmado será el signedSTC.xml.

Por último, una vez compuesto el CET firmado con las credenciales de CERTILOC, éste será devuelto vía HTTP a la entidad Sujeto.

La siguiente figura ilustra los lugares del documento XML unsignedSTC.xml que el módulo CERTILOC actualizará con la información espacial y temporal, y el resto del documento sirve a modo de plantilla:

```
- <saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:gml="http://www.opengis.net/gml" xmlns:sta="http://sta"
    saml:ID="_b00620e890b482e50e60b23cbcfdfb90"
    saml:IssueInstant="2012-10-16T15:34:27.000+0200" saml:Version="2.0">
    <saml:Issuer>urn:certiloc:issuer</saml:Issuer>
  + <saml:Subject>
    <saml:Conditions saml:NotBefore="2012-10-16T15:34:27.000+0200"
      saml:NotOnOrAfter="2012-10-16T15:34:27.000+0200"/>
  - <saml:AttributeStatement>
    - <saml:Attribute saml:FriendlyName="LocationInfo"
        saml:Name="urn:LocationInfo:attrib"
        saml:NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
      - <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">
        - <sta:LocationValue>
          - <gml:Point srsName="LL/WGS-84">
              <gml:pos>40.333255 -3.767046</gml:pos>
            </gml:Point>
          </sta:LocationValue>
        </saml:AttributeValue>
        <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">Arc Width:532 meters</saml:Attr
      </saml:Attribute>
    - <saml:Attribute saml:FriendlyName="TimeInfo" saml:Name="urn:TimeInfo:attrib"
        saml:NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri">
      - <saml:AttributeValue xmlns:xs="http://www.w3.org/2001/XMLSchema"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="xs:string">
        - <sta:TimeValue>
          - <gml:TimeInstant>
              <gml:timePosition>2012-10-16T15:34:27.000+0200</gml:timePosition>
            </gml:TimeInstant>
```

5.4 Problema 4: Falta de semántica de los documentos generados

5.4.1 Prefijos de espacios de nombres

Documento SpatialTemporalStamp.xml. Añadido prefijo y espacio de nombres a el nodo SpatialTemporalStamp del Sello Espacio Temporal. Esto es necesario para la firma, todo nodo debe tener identificado un espacio de nombres:

SITUACIÓN ANTERIOR:

```
-<SpatialTemporalStamp>
```

MODIFICACIÓN:

```
-<tsp:SpatialTemporalStamp
   xmlns:tsp="http://www.esat.kuleuven.ac.be/~kwouters/2002/08/xmltsp#">
```

Tambien se ha añadido el prefijo de espacio de nombres a otros nodos de los documentos de Petición de Sello de Tiempo (TimeStampRequest) y demás.

5.4.2 Modificación de los nodos File y Certificate

Documento SpatialTemporalStamp.xml. Modificados los nodos File y Certificate que referencian a el Mensaje y el Certificado Espacio Temporal respectivamente. Antes almacenábamos el Base 64 del contenido de cada archivo sin especificar el nombre del archivo. La solución propuesta ha sido referenciar el nombre del archivo correspondiente y además no poner su contenido tal cual ni en Base 64, sino dentro de una estructura DigestAlgValue que nos permite poner el resumen de su contenido como ilustra la siguiente figura:

El incluir el fichero en base 64 presenta una vulnerabilidad ya que desprovee este servicio de la propiedad de confidencialidad, al existir la posibilidad de que un tercero incercepte este mensaje para conocer el hecho de la acción de Sellado de cierto documento. Dicho de otra manera, en esta situación alguien puede conocer el hecho de que el Sujeto ha solicitado sellado espacio temporal de cierto documento.

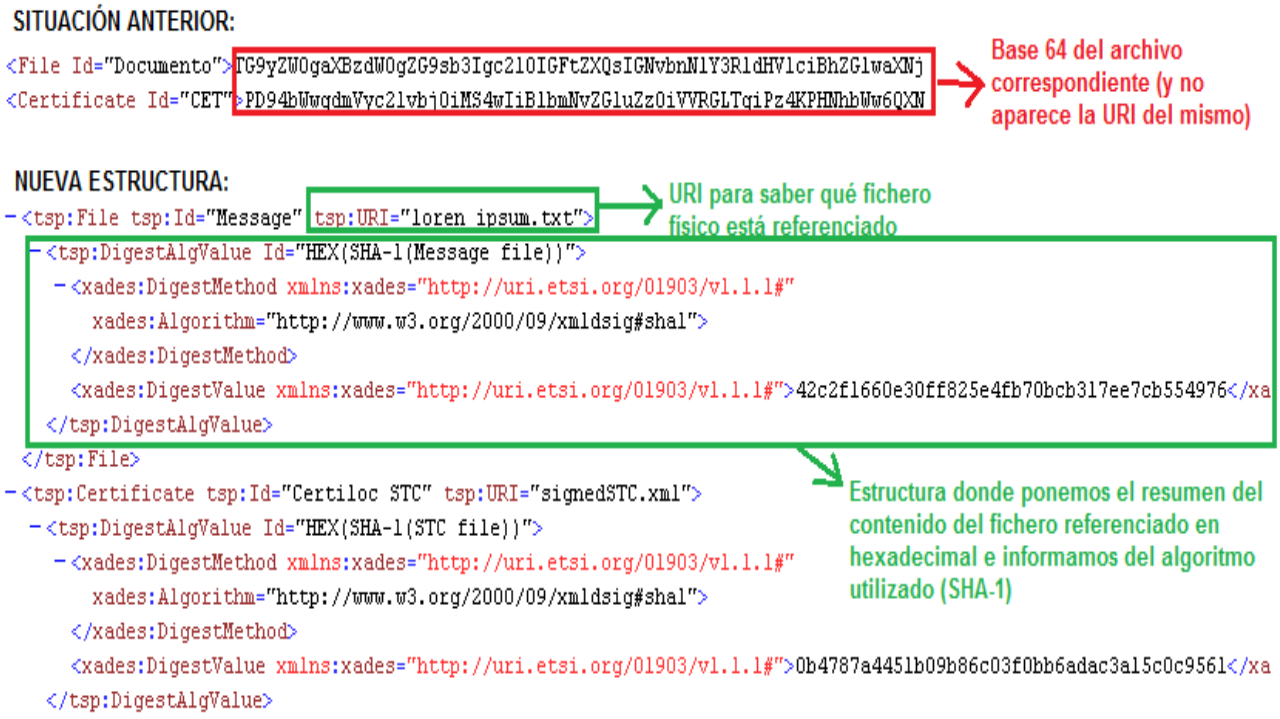


Figura 5.4 Modificación semántica de ficheros documento y CET en Sello Espacio Temporal

5.4.3 Uso del nodo tsp:References

La siguiente figura representa el XML Schema propuesto en el Anexo E de la tesis [2] que contiene el *lenguaje de especificación del protocolo de sellado temporal y de los ST*.

```
<xs:complexType name='TimeStampTokenType'>
<xs:sequence>
<xs:element ref='tsp:References' minOccurs='0'/>
<xs:element ref='tsp:MessageImprints' minOccurs='0'/>
<xs:element name='TSTInfo' type='tsp:TSTInfoType'/>
<xs:element ref='ds:Signature' minOccurs='0'/>
<xs:element ref='tsp:BindingInfo' minOccurs='0'/>
</xs:sequence>
</xs:complexType>
<xs:element name='References'>
<xs:complexType>
<xs:choice>
<xs:element ref='ds:Reference' maxOccurs='unbounded'/>
<xs:element name='XADESInfoLink'>
<xs:complexType>
<xs:attribute name='idref' type='xs:IDREF' use='required'/>
</xs:complexType>
</xs:element>
</xs:choice>
```

Como puede observarse el elemento de documento TimeStampToken contiene un elemento tsp:References cuya presencia erróneamente se omitió en el proyecto en el que nos basamos ([1]).

Este elemento es de importancia en el protocolo de Sellado Espacio Temporal por dos motivos: El primero es que semánticamente aporta la información que permite “enlazar” las dos partes principales del SET: la parte en la que la TSA mediante su firma electrónica acredita “el documento X a la fecha y hora T” con la otra parte en la que el Sujeto mediante su firma electrónica acredita “el documento X”.

El segundo motivo es que dicho elemento tsp:References, aporta seguridad, por el siguiente motivo:
es deseable que exista un mecanismo para evitar el envío directo del resúmen del mensaje a sellar temporalmente. Este mecanismo es el mencionado tsp:References, cuyo funcionamiento está explicado en detalle en la sección 6.1.5 Composición del Message Imprints de la TimeStamp Request

Lo que hacemos mediante el nodo tsp:References es que a la TSA le enviamos, en la TimeStampRequest – nodo "MessageImprints" (que es lo que la TSA entenderá por "documento a sellar temporalmente") un resúmen de dicho nodo tsp:References. De esta manera la TSA solo sabe que le han enviado para acreditar temporalmente un "resumen de una referencia" garantizando entonces confidencialidad.

En el momento de la construcción del Sello Espacio Temporal a partir de la respuesta de la TSA (TimeStampResponse) y Sigma, se incluye el nodo references que contiene un resumen de la firma de Sigma. y el Sujeto no podrá repudiar la evidencia del documento asociado a ese sello temporal porque la referencia contiene un resúmen de la firma de dicho documento (que llamamos Sigma)

5.5 Problema 5: Falta de confianza en la fuente de información temporal

Este problema ya fue identificado como una línea futura a seguir en el proyecto de partida ([1]). En concreto lo que se especifica es lo siguiente, en el capítulo 6.2 Líneas Futuras del mismo:

Confianza en la fuente de información temporal. Como se ha comentado, la información temporal se obtiene actualmente del sistema y es, por tanto, sensible a manipulaciones. Debe avanzarse en este aspecto e incluir en el demostrador de CERTILOC una fuente confiable de tiempo para la emisión de sellos temporales.

Continuando con lo explicado en el párrafo anterior, la hora del sistema puede estar desfasada con respecto a la zona horaria correcta, horario de verano, reseteada por reinstalaciones de equipo, caídas de corriente, manipulada malintencionadamente, etc.

Por estos motivos se ha buscado un protocolo o técnica que permitiera proporcionar una fuente de información temporal fiable. El protocolo NTP cumple con este propósito como se especifica en [21]: *Network Time Protocol (NTP) is a networking protocol for clock synchronization between computer systems over packet-switched, variable-latency data networks.*

En este proyecto implementamos el acceso a una Fuente Confiable de Tiempo mediante el protocolo NTP versión 3, entendiéndose por tal aquella especificada en la RFC 1305.

Aunque ya hay en curso una versión NTP versión 4, dicha versión, especificada en RFC 5905, está aún en estado de proposed standard y no hay implementaciones disponibles (al menos de dominio público) para la misma, por lo que en este proyecto utilizamos la versión 3. Para tal efecto utilizaremos alguna de las diversas librerías externas java existentes.

La TSA devuelve la información de tiempo en el nodo Gentime, que es un subnodo de TimeStampToken/TSTInfo. Como se puede apreciar en la siguiente figura:

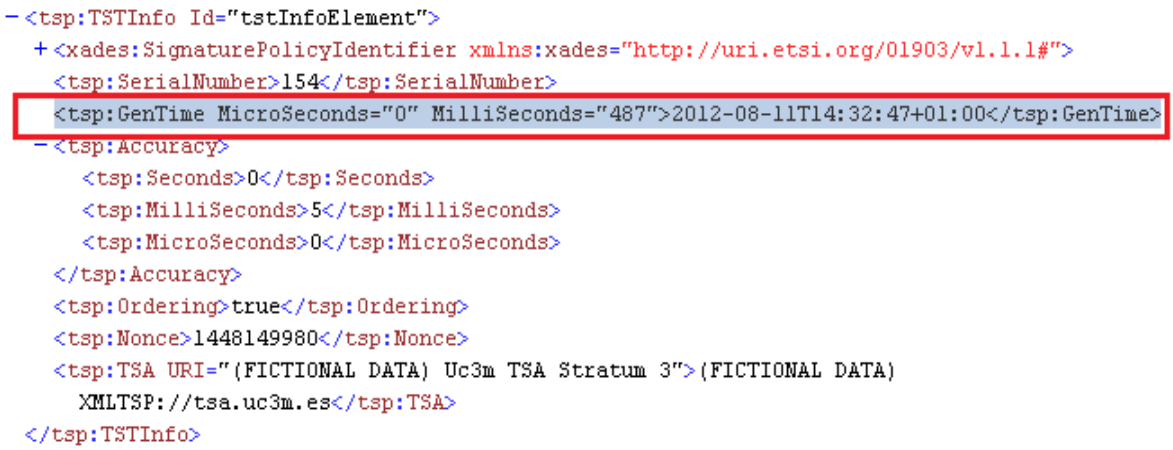


Figura 5.5: Situación de la información de tiempo de la TSA en el xml

El objetivo es, por lo tanto, realizar las modificaciones necesarias en el código de aplicación que realiza la construcción del nodo TSTInfo del documento de Respuesta de Sello de Tiempo (TimeStamp Response).

Por otro lado, se busca incluir en nuestra implementación una manera de que, en caso de que el protocolo NTP sea inaccesible por motivos de conectividad de red, bloqueo de puertos, firewalls, etc, se disponga de un método alternativo de obtención del tiempo. Se ha decidido dos métodos alternativos y configurables desde las propiedades de Configuración: uno consiste en obtener la fecha del sistema donde esté instalada la Autoridad de Sellado Temporal (TSA), el otro consiste en poder fijar una fecha para las sucesivas peticiones de Sello de Tiempo.Se ha tenido en consideración y se ha comprobado que el horario de verano se ve reflejado en la fecha y hora, esto se ha comprobado en el cambio de hora que hubo el 28/10/2012 que pasamos de UTC+2 a UTC+1.

5.6 Problema 6: Carencia de una estructura suficiente de entidades de certificación

5.6.1 Situación previa

Como se explicó en la sección 4. ANÁLISIS DEL PROBLEMA Y ALTERNATIVAS DE SOLUCIÓN, hay una falta de definición de una estructura de certificación suficientemente ilustrativa de una aplicación real del protocolo del presente proyecto. Partimos de una situación en la que solo disponemos de dos entidades autofirmadas, una para el sujeto y otra para la tsa. Además estas entidades tienen un certificado de clave pública emitido en la versión V1 del estándar X509, que se considera obsoleta.

Por otro lado en las firmas ds:signature de los xmls a tratar, es decir las firmas generadas por un lado por el sujeto y por otro lado la tsa, tienen cada una un nodo KeyInfo con un solo certificado. En una situación real debería ser un certificado emitido por una Autoridad de Certificación reconocida que a su vez puede ser una AC intermedia de otra AC "padre".

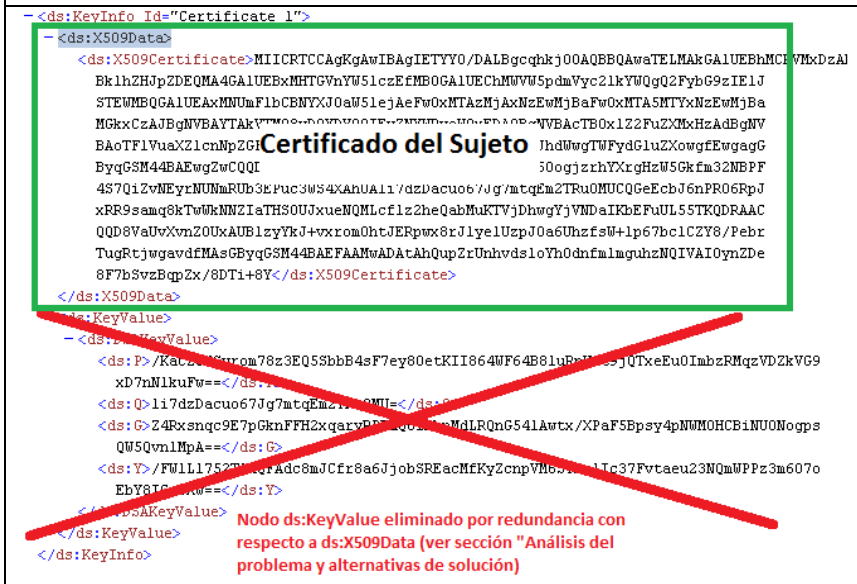
5.6.2 Análisis de la solución

Lo que se propone es crear una nueva estructura de certificación ayudados por una herramienta como openssl para crear los nuevos almacenes de claves y certificados de clave pública.
En las figuras 1 y 2 podemos apreciar la diferencia de certificación entre la versión de partida y el nuevo esquema de certificación propuesto.

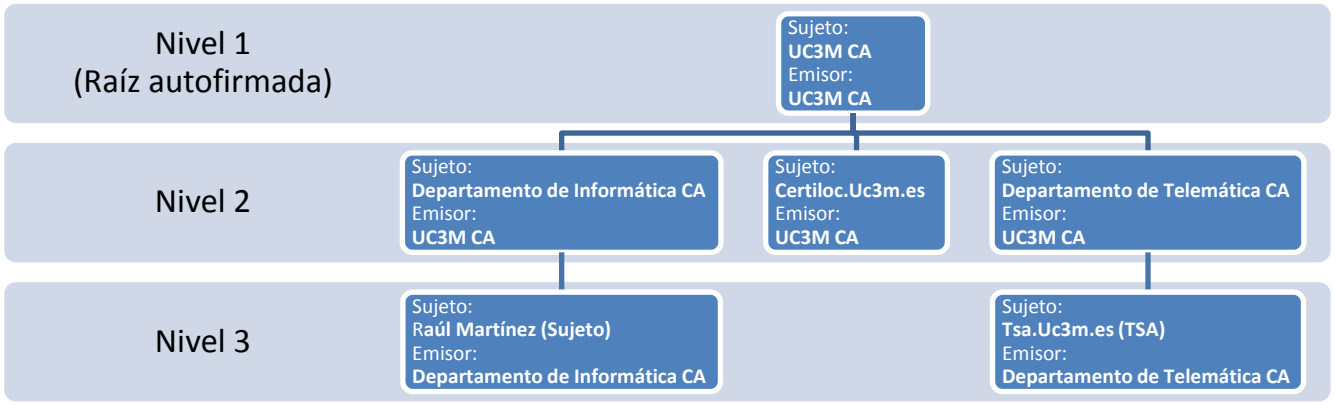
Figura 1: En esta firma (en este caso es la firma del sujeto) solo tenemos un nodo X509Data que contiene un único certificado X509Certificate. Por otro lado, encontramos una redundancia en el elemento ds:KeyValue, por lo que este se puede eliminar. Esto queda demostrado en las especificaciones del estándar XMLDSIG, donde se ve que el tipo KeyInfoType es un choice entre X509Data , KeyValue y otros 5 posibles elementos. La figura siguiente ilustra este hecho:

```
<element name="KeyInfo" type="ds:KeyInfoType"/>
<complexType name="KeyInfoType" mixed="true">
  <choice maxOccurs="unbounded">
    <element ref="ds:KeyName"/>
    <element ref="ds:KeyValue"/>
    <element ref="ds:RetrievalMethod"/>
    <element ref="ds:X509Data"/>
    <element ref="ds:PGPData"/>
    <element ref="ds:SPKIData"/>
    <element ref="ds:MgmtData"/>
    <any processContents="lax" namespace="##other"/>
    <!-- (1,1) elements from (0,unbounded) namespaces -->
  </choice>
  <attribute name="Id" type="ID" use="optional"/>
</complexType>
```

Figura 2: Vemos que tenemos varios nodos ds:X509Certificate. El primero es el firmante, a continuación va la AC del firmante y por último va la AC raíz. La redundancia que mencionamos en la figura 1 la hemos resuelto eliminando el elemento ds:KeyValue de la generación de la firma.

SITUACIÓN DE PARTIDA CON UN UNICO CERTIFICADO POR FIRMA	NUEVO ESQUEMA DE FUNCIONAMIENTO PROPUESTO CON UNA RUTA DE CERTIFICACION PARA EL FIRMANTE
	

5.6.3 Estructura de certificación propuesta



Información que se usará para generar los certificados de las entidades mediante la herramienta escogida (openssl, escogida por su potencia y versatilidad):

Uc3mCA.root.crt
SUJETO (SUBJECT):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
CN = CA raíz Universidad Carlos III
Flag "Root CA": verdadero.
EMISOR(ISSUER):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
CN = CA raíz Universidad Carlos III

DtoInformaticaCA.crt
SUJETO (SUBJECT):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
OU = Departamento Informática
CN = CA Departamento Informatica
EMISOR(ISSUER):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
CN = CA raíz Universidad Carlos III

CERTILOC.Uc3m.es.crt
SUJETO (SUBJECT):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
OU = Dto Sistemas Información
CN = CERTILOC.Uc3m.es
EMISOR(ISSUER):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
CN = CA raíz Universidad Carlos III

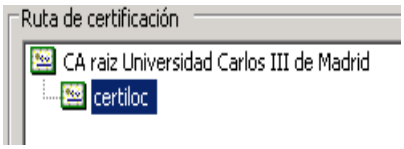
DtoTelematicaCA.crt
SUJETO (SUBJECT):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
OU = Departamento Telemática
CN = CA Departamento Telemática
EMISOR(ISSUER):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
CN = CA raíz Universidad Carlos III

Subject.crt
SUJETO (SUBJECT):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
OU = Departamento Informática
CN = Raul Martinez
EMISOR(ISSUER):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
OU = Departamento Informática
CN = CA Departamento Informatica

TSA.Uc3m.es.crt
SUJETO (SUBJECT):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
OU = Departamento Telemática
CN = tsa.uc3m.es
EMISOR(ISSUER):
C = ES
ST = Madrid
L = Leganes
O = Universidad Carlos III
OU = Departamento Telemática
CN = CA Departamento Telemática

La estructura que acabamos de describir es fija, es decir, que sin dificultad podemos cambiar los certificados de clave pública de cada una de las 6 entidades descritas, pero para añadir elementos a la estructura de certificación eso requiere seguir las instrucciones descritas en la Sección Apéndice Manual de Mantenimiento.

Estos son los certificados una vez creados:

Certificado del Sujeto (Subject.crt)	Certificado de CERTILOC (CERTILOC.Uc3m.es.crt)	Certificado de la TSA (TSA.Uc3m.es.crt)
		

5.7 Problema 7: Incorrecto tratamiento del Nonce

El nonce es un número arbitrario utilizado solo una vez en una comunicación criptográfica. Con frecuencia es un número aleatorio o pseudo-aleatorio emitido en protocolos de autenticación para asegurar que antiguas comunicaciones no puedan ser utilizadas en ataques de Replay (también llamados "ataques de reinyección").

En el ámbito del presente proyecto se utiliza un nonce para cada generación de Sello Espacio Temporal. En concreto, se utiliza para la parte relativa a la información temporal (Sello de Tiempo), porque para obtener dicha información recurrimos al Servicio de Sellado de Tiempo proporcionado por la Autoridad de Sellado Temporal (TSA).

La obtención de dicho Sello de Tiempo requiere de una comunicación entre el Sujeto, que hace de parte solicitante y la TSA, que emite dicho Sello. El funcionamiento consiste en que el Sujeto envía a la TSA un documento XML llamado documento de Petición de Sello de Tiempo(TimeStamp Request), que contendrá un número nonce generado aleatoriamente. El funcionamiento ideal es que la TSA procese el documento de Petición de Sello de Tiempo extrayendo dicho nonce y devolviéndolo junto con la información temporal en el documento de Respuesta de Sello de Tiempo (TimeStamp Response).

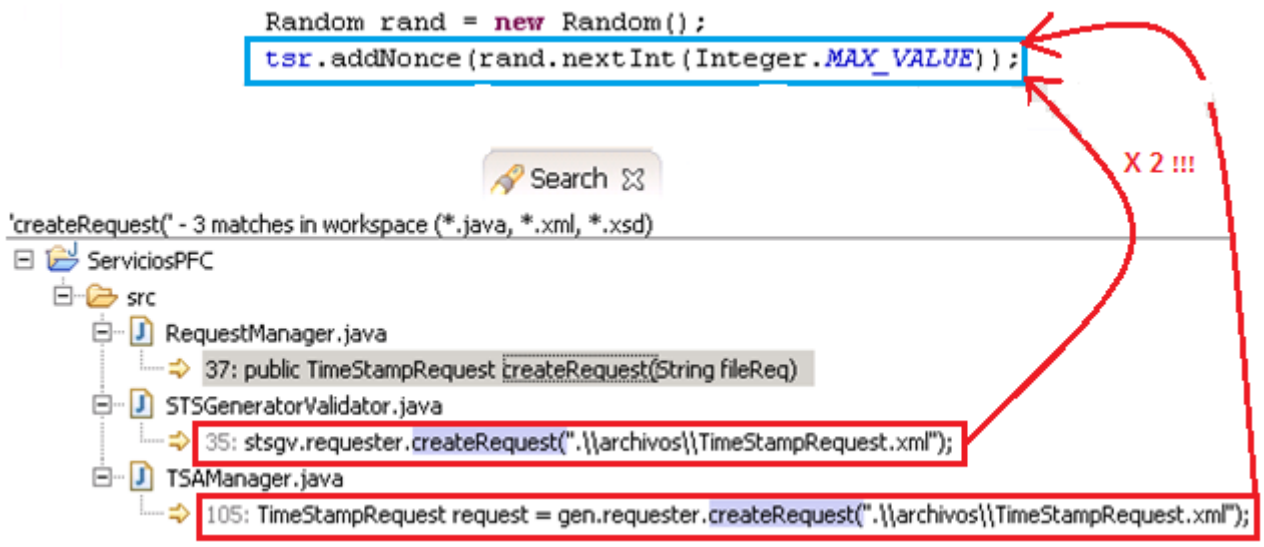
5.7.1 Análisis del problema: situación del software de partida

El software presentado en el proyecto [1] incumple el funcionamiento del nonce explicado en el párrafo anterior, al generarlo dos veces, una al componer el documento de Petición de Sello de Tiempo y la otra al componer el documento de Respuesta de Sello de Tiempo.

Podemos justificar este hecho a dos niveles: por un lado tenemos los documentos xml donde aparece el nonce distinto para un único escenario de generación de Sello Espacio Temporal, y por otro lado podemos demostrarlo a nivel de código.

La siguiente figura, fruto de una comprobación depurando código, refleja que a nivel de documento tenemos distintos nonce:

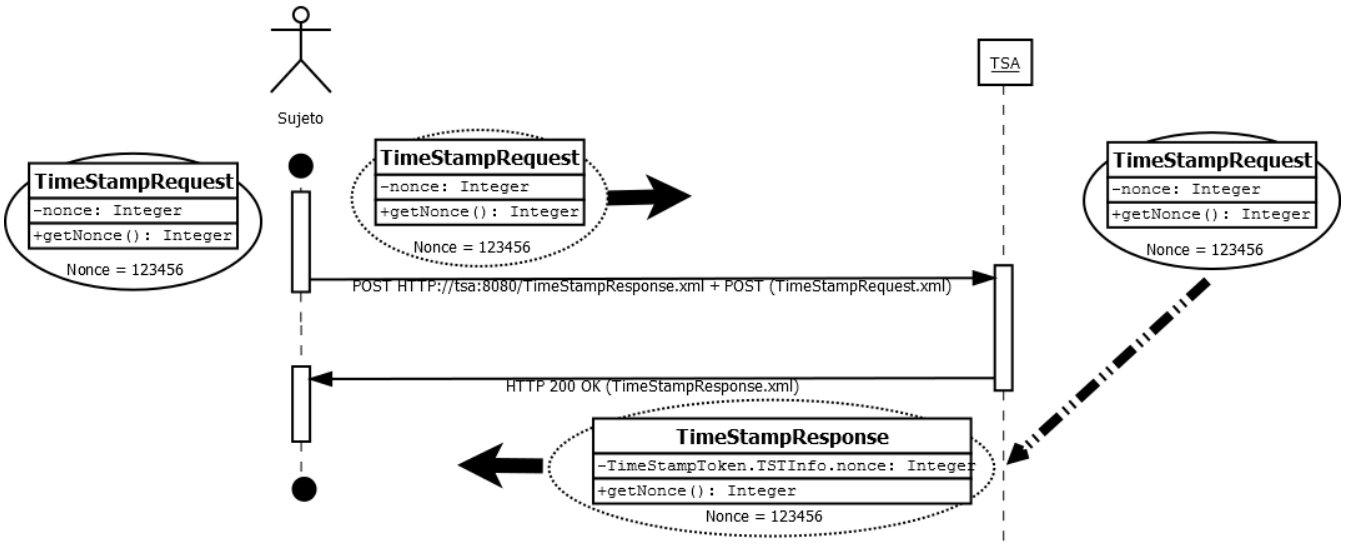
A nivel de código, como ilustra la siguiente figura, se pueden ver los dos puntos donde se invoca al método CreateRequest de la clase RequestManager . Puesto que en el método CreateRequest se calcula aleatoriamente el nonce mediante la instrucción nextInt, el resultado es que el nonce se genera dos veces en distintos puntos del código.



5.7.2 Solución aportada

La manera de resolver este problema ha sido que gracias al conjunto de refactorizaciones de código, la clase TimeStampRequest se instancia en la TSA a partir de la petición de Sello de Tiempo recibida por la TSA, pudiendo acceder al nonce que generó el Sujeto al crear la petición de Sello por primera vez.

El conjunto de refactorizaciones del código de aplicación ha implicado que la clase TimeStampRequest (petición de sello de tiempo) ahora solamente se instancia en los momentos que realmente se requiere. A efectos de cálculo del nonce, en la TSA cuando se trata la petición recibida se obtiene dicho nonce y se recuerda para despues devolver ese preciso valor en el documento de respuesta de Sello de Tiempo.



5.8 Problema 8: Ausencia de validación de los certificados

Este problema ya fue identificado como una línea futura a seguir en el proyecto de partida ([1]). En concreto lo que se especifica es lo siguiente, en el capítulo 6.2 Líneas Futuras del mismo:

Validez de información de clave pública: Debe implementarse una forma de asegurar que los certificados de clave pública empleados son válidos en el momento de su uso.

El proyecto en el que nos basamos ([1]) implementa una funcionalidad de validación cuyo objeto es validar las firmas XML Signature incluídas en los documentos XML que integran el Sistema de Sellado Espacio Temporal, sin embargo dicha funcionalidad adolece de una falta de validación de la información de clave pública que se utiliza para validar dichas firmas. La información de clave pública viene en formato de Certificados X.509, y estos certificados en sí mismos necesitan ser validados también, como ilustra la siguiente figura con un ejemplo de XML Signature (nodo ds:Signature).

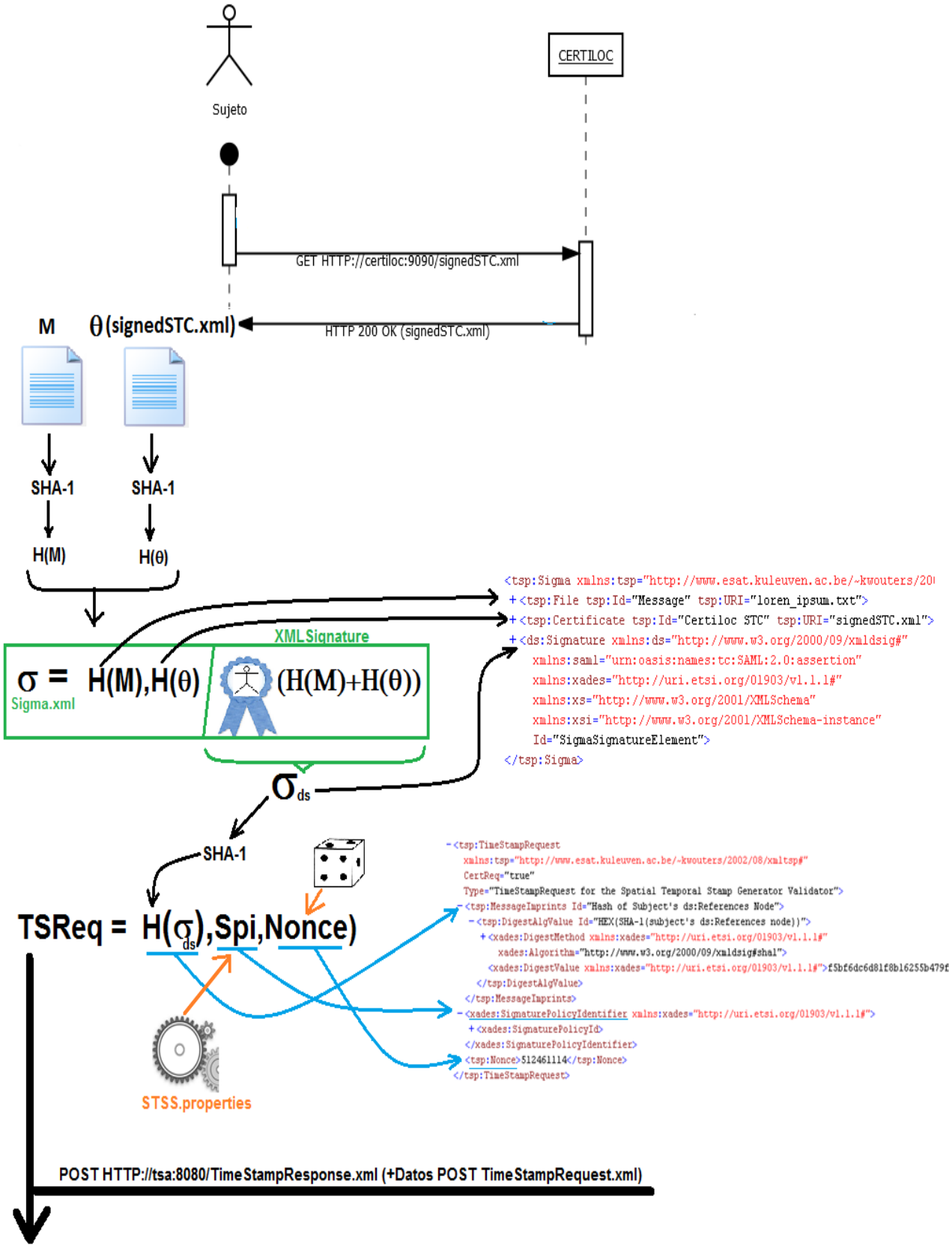
Figura 5.8 Situación de la información de clave pública a validar en el SET

En la figura anterior se aprecia que existe el nodo ds:X509Data, que es el contenedor directo de los certificados X509. Como explicamos en la sección 5.6 Problema 6: Carencia de una estructura suficiente de entidades de certificación, en el presente proyecto se contempla que para cada firma XMLSignature la información de clave pública venga dada por una cadena de certificados en lugar de solo un certificado como pasaba en el proyecto de partida ([1]).

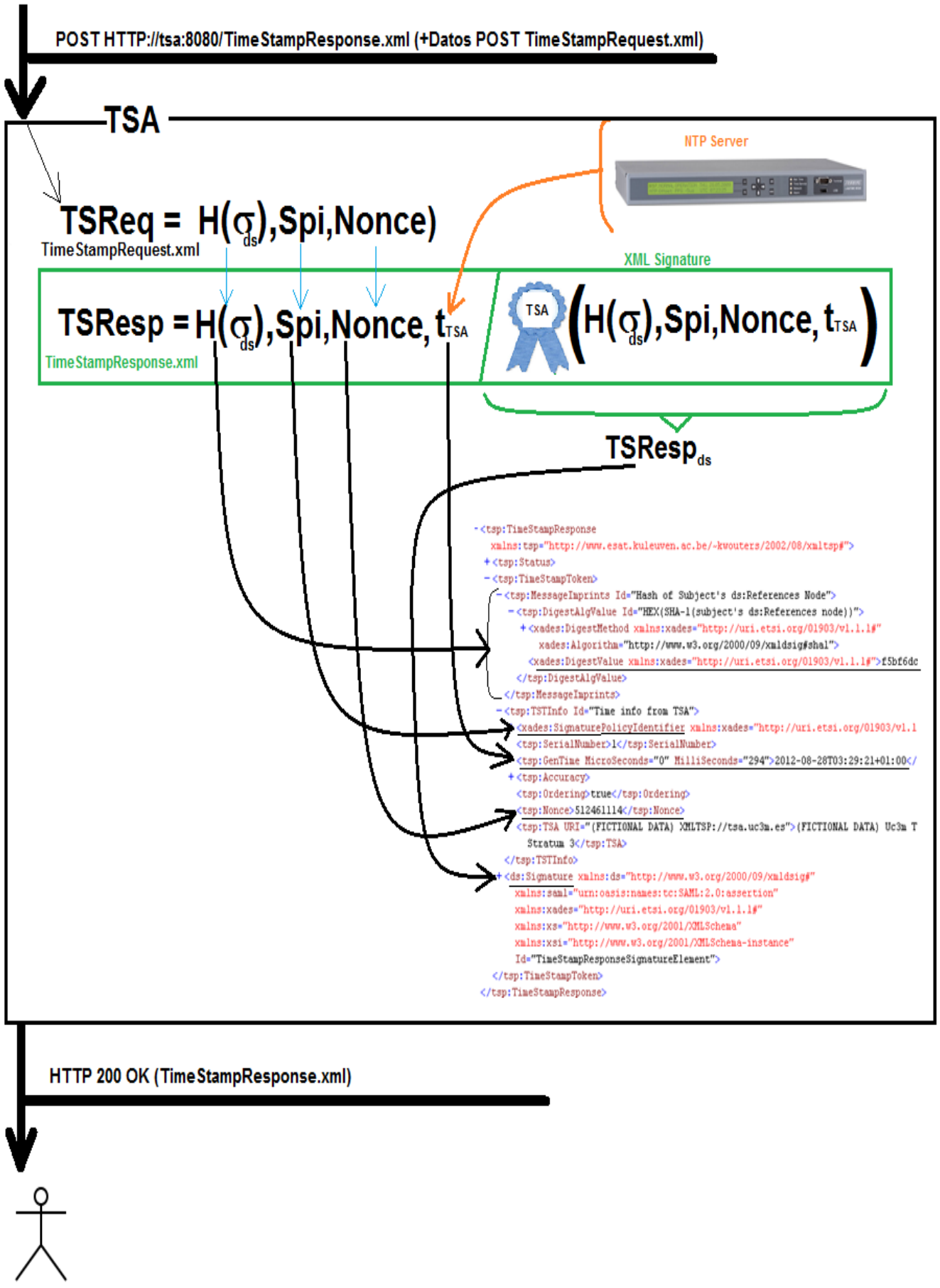
Se creará un método llamado validateCertificateChain que reciba como parámetros: a) el nombre del documento XML que contiene la firma XMLSignature y b) una expresión (se sugiere una expresión XPath) que describa la localización del nodo XMLSignature en dicho documento XML.

5.9 Proceso de creación del Sello Espacio Temporal (alto nivel)

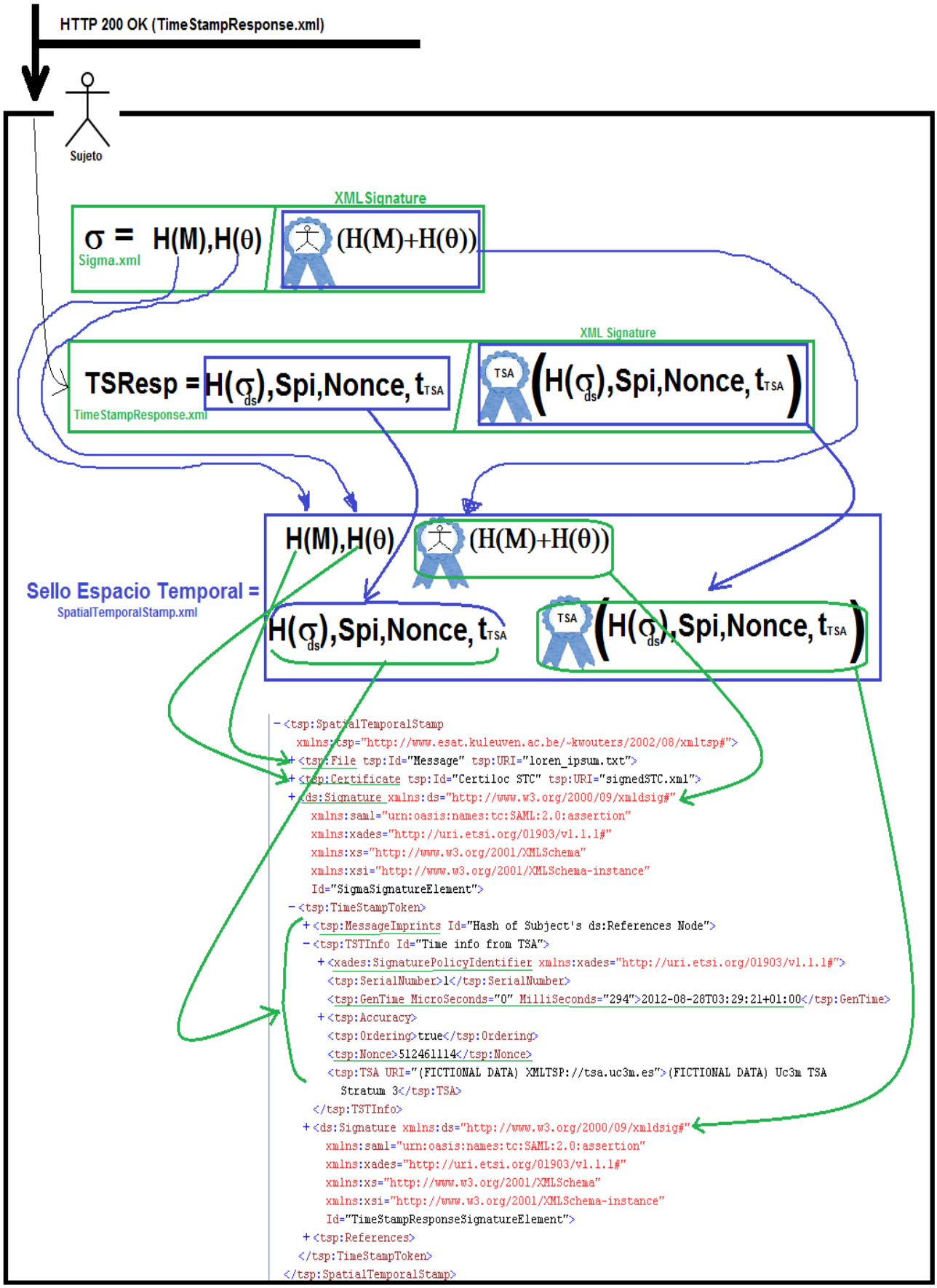
El Sujeto toma el Mensaje y el Certificado Espacio Temporal y compone Sigma (firmándolo con su clave privada). Con éste Sigma tomando la Signature Policy del archivo de configuración, y el Nonce generado aleatoriamente, compone la TimeStampRequest, que envía por http a la TSA solicitándole una respuesta (TimeStampResponse):



La TSA procesa la petición del Sujeto (TimeStampRequest) añadiéndole la información de tiempo sacada de un servidor NTP, firma el conjunto y lo devuelve como TimeStampResponse vía http al Sujeto:



El Sujeto toma la respuesta de la TSA (TimeStampResponse) y la combina con la entidad Sigma que creó al principio, la entidad resultante es el Sello Espacio Temporal.



6. IMPLEMENTACIÓN

6.1 Creación del Sello Espacio Temporal

6.1.1 Composición del resumen del Mensaje de Sigma

Al instanciar un objeto de la clase Message indicándole el nombre del fichero Mensaje, se calcula el resumen en hexadecimal de dicho fichero y se guarda en un nodo DigestAlgValue

```
- <tsp:File tsp:Id="Message" tsp:URI="loren_ipsum.txt">
  - <tsp:DigestAlgValue Id="HEX(SHA-1(Message file))">
    - <xades:DigestMethod xmlns:xades="http://uri.etsi.org/01903/v1.1.1#"
        xades:Algorithm="http://www.w3.org/2000/09/xmldsig#sha1">
      </xades:DigestMethod>
      <xades:DigestValue xmlns:xades="http://uri.etsi.org/01903/v1.1.1#">42c2f1660e30ff825e4fb70bcb317ee7cb554976</xades:DigestValue>
    </tsp:DigestAlgValue>
</tsp:File>
```

```
    public Message(String sMessageFile) throws Exception

        //Compute hash of message file
        String sDigestAlgorithm = MessageDigestAlgorithm.ALGO_ID_DIGEST_SHA1;

        this.sMessageHash = computeHash(sMessageFile);

        hashOfMessage = new DigestAlgValue(
                sDigestAlgorithm,
                sMessageHash,
                "HEX(SHA-1(Message file))");

        Id = Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DEFAULTMESSAGENODE.ID");

        Uri =   sMessageFile;

    }
```

Obtenemos los bytes del fichero del Mensaje y calculamos el hash pasándolo a hexadecimal

Por defecto es lorem_ipsum.txt

Message

6.1.2 Composición del resumen del Certificado Espacio Temporal en Sigma

Al instanciar un objeto de la clase SpatialTemporalCertificate indicándole el nombre del fichero del Certificado Espacio Temporal, se calcula el resumen en hexadecimal de dicho fichero y se guarda en un nodo DigestAlgValue

```
-<tsp:Certificate tsp:Id="Certiloc_STC" tsp:URI="signedSTC.xml">
  -<tsp:DigestAlgValue Id="HEX(SHA-1(STC file))">
    -<xades:DigestMethod xmlns:xades="http://uri.etsi.org/01903/v1.1.1#"
       xades:Algorithm="http://www.w3.org/2000/09/xmldsig#sha1">
     </xades:DigestMethod>
     <xades:DigestValue xmlns:xades="http://uri.etsi.org/01903/v1.1.1#">0b4787a4451b09b86c03f0bb6adac3a15c0c9561</xades:DigestValue>
   </tsp:DigestAlgValue>
 </tsp:Certificate>
```

```
public SpatialTemporalCertificate(String sSpatialTemporalCertificateFile) throws Exception
{
    //Compute hash of stc file
    String sDigestAlgorithm = MessageDigestAlgorithm.ALGO_ID_DIGEST_SHA1;
    this.sStcHash = computeHash(sSpatialTemporalCertificateFile);
    hashOfStc = new DigestAlgValue(
            sDigestAlgorithm,
            sStcHash,
            "HEX(SHA-1(STC file))");

    Id = Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DEFAULTSTCNODE.ID");

    Uri =   sSpatialTemporalCertificateFile;

}
```

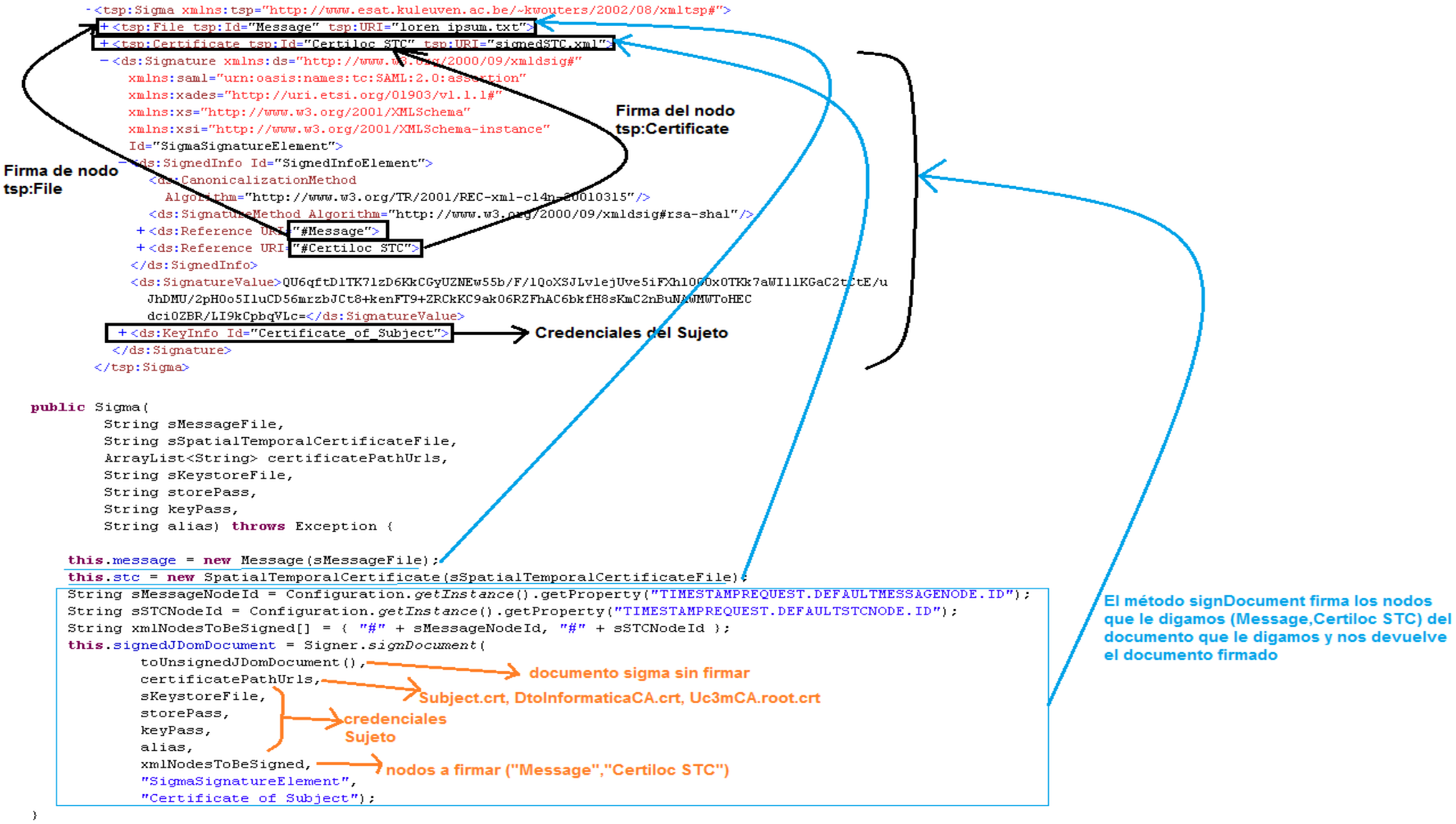
Obtenemos los bytes del fichero del Certificado Espacio Temporal y calculamos su resumen pasándolo a hexadecimal

Certiloc STC

Por defecto es signedSTC.xml

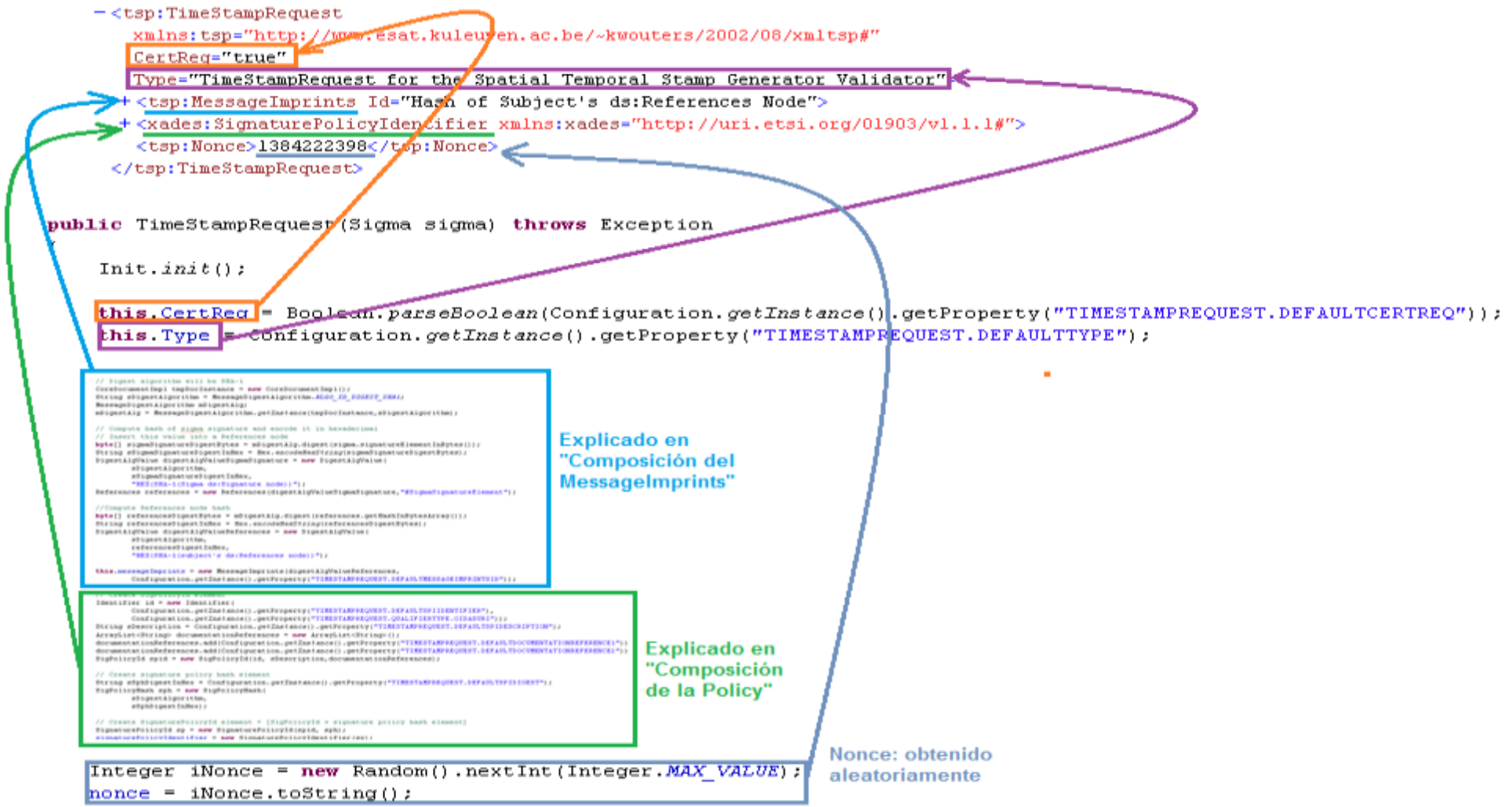
6.1.3 Composición y firma de Sigma

El sujeto compone Sigma firmado, que contiene los nodos tsp:File (el mensaje) y tsp:Certificate (el certificado e-t de certiloc) firmados. Este Sigma inicialmente se guarda en un fichero Sigma.xml

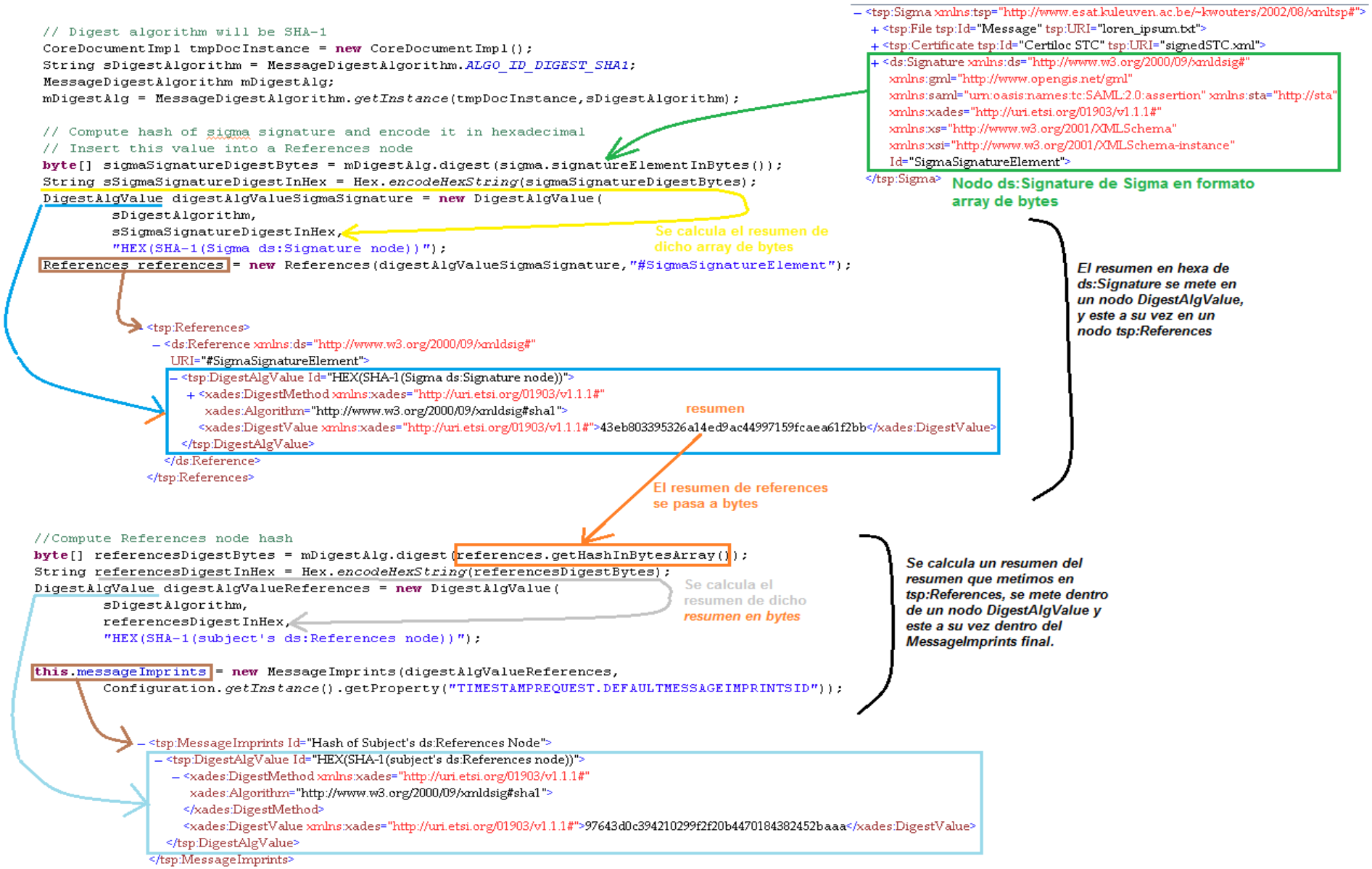


6.1.4 Composición de la petición de sello de tiempo (TimeStamp Request)

CertReq, Type y el Nonce se obtienen (el nonce de manera aleatoria), y el MessageImprints y el SignaturePolicyIdentifier se explican en las secciones" 4.2.5 Composición del Message Imprints de la TimeStamp Request" y "4.2.6 Composición de la Policy de la TimeStamp Request"



6.1.5 Composición del Message Imprints de la TimeStamp Request



6.1.6 Composición de la Policy de la TimeStamp Request

Los valores de este nodo son fijos, aunque en el software de este Proyecto se han parametrizado como propiedades de configuración:

```
- <xades:SignaturePolicyIdentifier xmlns:xades="http://uri.etsi.org/01903/v1.1.1#">
  - <xades:SignaturePolicyId>
    - <xades:SigPolicyId>
        <xades:Identifier Qualifier="OIDAsURI">(FICTIONAL DATA)
          http://www.uc3m.es/DtoInformatica/DocInterna/STSPolicy.odt</xades:Identifier>
        <xades:Description>This is the policy for the STS system.</xades:Description>
      - <xades:DocumentationReferences>
          <xades:DocumentationReference>STS User Manual</xades:DocumentationReference>
          <xades:DocumentationReference>STS Technical Specifications</xades:DocumentationReference>
        </xades:DocumentationReferences>
      </xades:SigPolicyId>
    - <xades:SigPolicyHash>
        <xades:DigestMethod xades:Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
        <xades:DigestValue>(FICTIONAL DATA)
          WFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhYWFhY
      </xades:SigPolicyHash>
    </xades:SignaturePolicyId>
  </xades:SignaturePolicyIdentifier>


  // Create SigPolicyId element
  Identifier id = new Identifier(
          Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DEFAULTSPIIDENTIFIER"),
          Configuration.getInstance().getProperty("TIMESTAMPREQUEST.QUALIFIERTYPE.OIDASURI"));
  String sDescription = Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DEFAULTSPIDESCRIPTION");
  ArrayList<String> documentationReferences = new ArrayList<String>();
  documentationReferences.add(Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DEFAULTDOCUMENTATIONREFERENCE1"));
  documentationReferences.add(Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DEFAULTDOCUMENTATIONREFERENCE2"));
  SigPolicyId spid = new SigPolicyId(id, sDescription,documentationReferences);

  // Create signature policy hash element
  String sSphDigestInHex = Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DEFAULTSPIDIGEST");
  SigPolicyHash sph = new SigPolicyHash(
          sDigestAlgorithm,
          sSphDigestInHex);

  // Create SignaturePolicyId element = [SigPolicyId + signature policy hash element]
  SignaturePolicyId sp = new SignaturePolicyId(spid, sph);
  signaturePolicyIdentifier = new SignaturePolicyIdentifier(sp);
```

6.1.7 Composición de la respuesta de la TSA (TimeStamp Response)

El constructor de TimeStampResponse recibe como parámetros las credenciales de la tsa, porque el nodo TimeStampToken irá firmado. La construcción del nodo TimeStampToken se explica en la sección 6.1.8 Composición del subnodo TimeStampToken de la TimeStamp Response

6.1.8 Composición del subnodo TimeStampToken de la TimeStamp Response

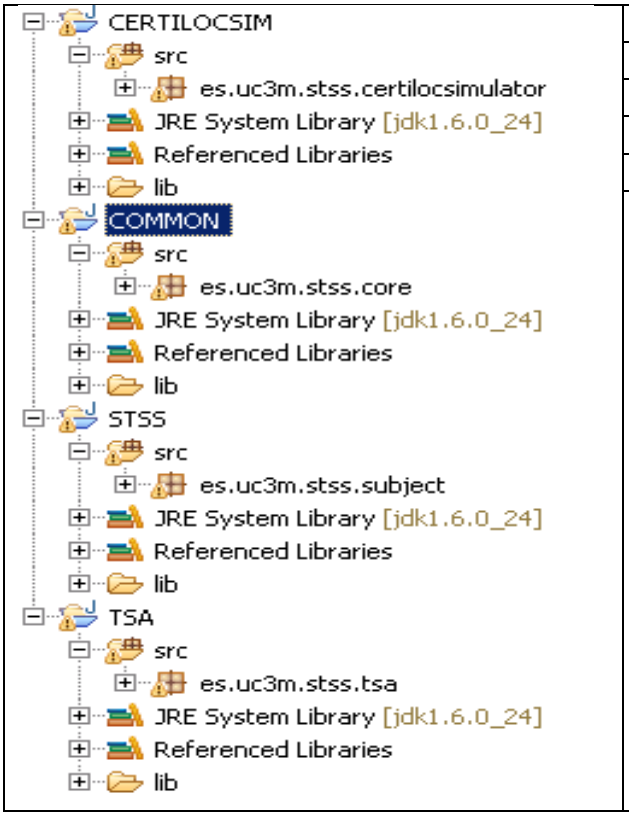
La TSA firma los nodos MessageImprints y TSTInfo y devuelve un documento TimeStampToken firmado:


```
public TimeStampToken(
        MessageImprints messageImprints,
        SignaturePolicyIdentifier spi,
        String Nonce,
        String sSerialNumber,
        ArrayList<String> certificatePathUrls,
        String store,
        String storePass,
        String keyPass,
        String alias)  throws Exception
        {

    this.messageImprints = messageImprints;
    this.tstInfo = new TSTInfo(spi, sSerialNumber, Nonce);
    String sMessageImprintsNodeId = Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DEFAULTMESSAGEIMPRINTSID");
    String sTstInfoNodeId = Configuration.getInstance().getProperty("TIMESTAMPRESPONSE.DEFAULTTSTINFO.ID");
    String xmlNodesToBeSigned[] = ( "#" + sMessageImprintsNodeId, "#" + sTstInfoNodeId );

    this.signedJDomDocument = Signer.signDocument(toUnsignedJDomDocument(),
            certificatePathUrls,
            store,
            storePass,
            keyPass,
            alias,
            xmlNodesToBeSigned,
            "TimeStampResponseSignatureElement",
            "Certificate_of_tsa");
        }
```

6.2 Arquitectura del Sistema

6.2.1 Estructura

El sistema se compone de cuatro módulos: tres módulos para las entidades principales (Sujeto, TSA, CERTILOC) y un módulo común que contiene las clases que tienen en común cada uno de los módulos

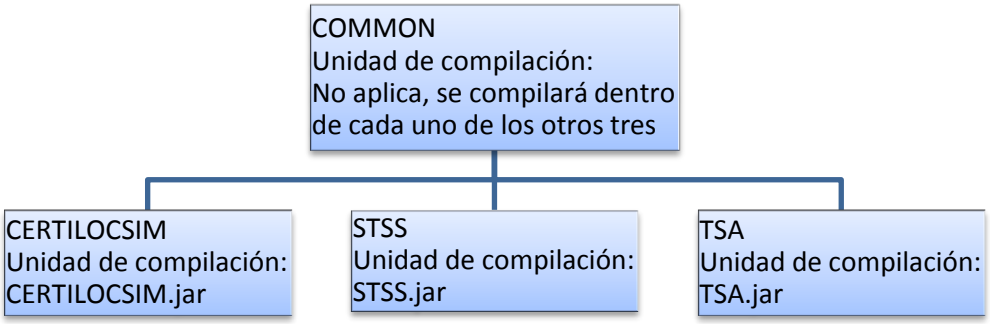


MÓDULO	NOMBRE DEL PAQUETE
CERTILOCSIM	Es.uc3m.stss.certilocsimulator
COMMON	Es.uc3m.stss.core
STSS	Es.uc3m.stss.subject
TSA	Es.uc3m.stss.tsa

6.2.2 Árbol de dependencias entre módulos

La arquitectura en módulos se ha aplicado para proporcionar independencia a los tres grandes grupos funcionales de la aplicación (Sujeto, TSA, CERTILOC). De tal manera, si modificamos uno de los tres módulos principales (CERTILOCSIM, STSS,TSA) habrá que generar el paquete correspondiente nada más.

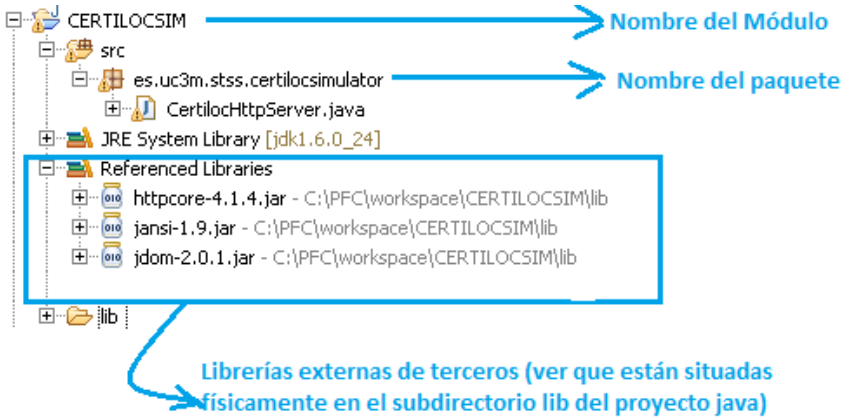
Por otro lado, los tres módulos principales (CERTILOCSIM, STSS,TSA) dependen del módulo común (COMMON), de manera que si modificamos el módulo común habrá que regenerar los paquetes jar correspondientes a cada uno de los otros tres módulos:



6.3 Módulo Simulador de CERTILOC

6.3.1 Visión general

Este módulo es el encargado de simular un servidor de CERTILOC encargado de servir Certificados Espacio Temporales.



El servidor se ha implementado como un servidor HTTP encargado de recibir una petición de archivo y devolver el Certificado Espacio Temporal como respuesta, como se explica en la sección 5.3 Problema 3: Falta de implementación del módulo CERTILOC.

La clase única de este módulo se llama CERTILOCHttpServer. Está basada en otra clase llamada ElementalHttpServer, descrita como un servidor básico de http 1.1 basado en un modelo de entrada/salida bloqueante (asíncrono). Dicha clase la hemos tomado de la documentación en línea de dicho componente HttpCore, con pocas modificaciones.

Esta clase se encarga de las siguientes operaciones:

- Servir peticiones http devolviendo el fichero requerido tomando como configuración los siguientes parámetros:
  - CERTILOCSIM.SERVER.ROOTDIRECTORYFORSERVINGFILES: Directorio por defecto del servidor
  - CERTILOCSIM.SERVER.PORT: Puerto en el que escuchar posibles peticiones
- Utilidad de firma de Certificado Espacio Temporal: Esta clase es capaz de firmar un Certificado Espacio Temporal con las credenciales que hemos definido para el simulador de CERTILOC:
  - Precondición: que el Certificado Espacio temporal exista y no haya sido firmado previamente.

6.3.2 Diagrama de clases

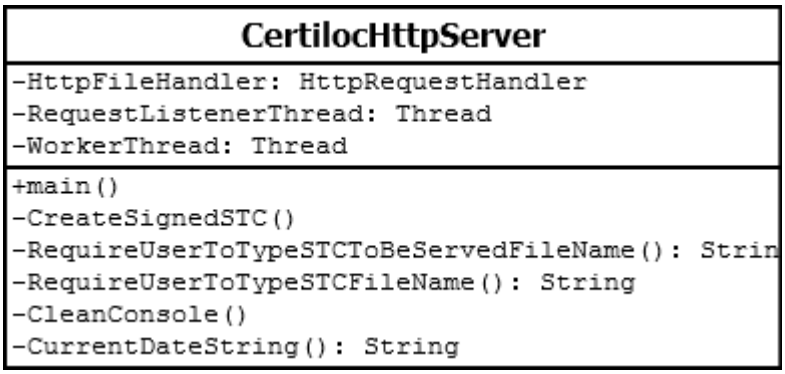


Figura 6.3.2 Diagrama de clases del módulo del Simulador de CERTILOC

6.3.3 Dependencias de librerías externas

Este módulo depende de las librerías siguientes:

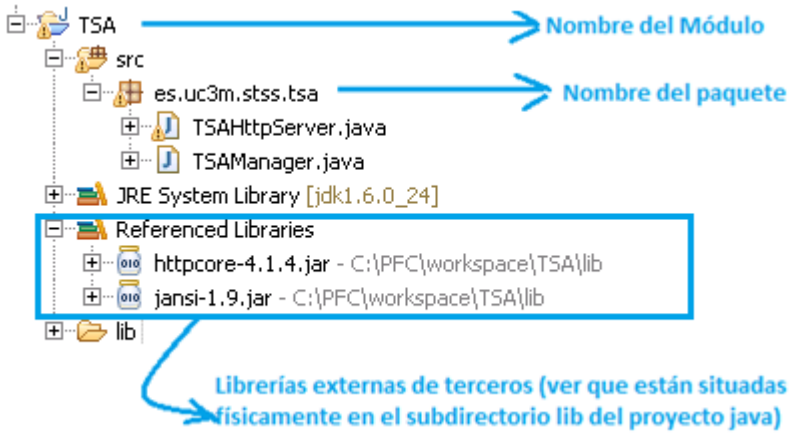
Nombre	Origen	Empaquetado	Propósito
Apache HttpComponents 4.1.4	http://hc.apache.org/	httpcore-4.1.4.jar	Hacer que CERTILOC sea un servidor http de Certificados Espacio Temporales
Jansi 1.9	http://jansi.fusesource.org/	jansi-1.9.jar	Para que la salida por consola sea con colores, se pueda borrar la pantalla, etc
JDOM 2.0.1	http://www.jdom.org/	jdom-2.0.1.jar	Permitir el manejo en memoria de documentos xml conforme al modelo de objetos de documento (DOM) con comodidad. En este ámbito sirve para permitir la utilidad de firma de Certificados Espacio Temporales de CERTILOC (ver manual de Administrador – utilidad de firma de CERTILOC)

6.4 Módulo de la TSA

6.4.1 Visión general

Este módulo es el encargado de recibir las peticiones de tiempo (Timestamp Requests) por parte del Sujeto y componer una respuesta (Timestamp Response) en base a dicha petición.

El código de este módulo está implementado en un proyecto java llamado TSA dentro del workspace de código, como se muestra en la siguiente figura:



Este módulo irá empaquetado dentro de un archivo jar
Este servicio se ha implementado como un servidor HTTP encargado de recibir una petición de archivo, en este caso una petición del archivo de respuesta Timestamp Response. Junto con la petición la TSA recibirá el Timestamp Request como datos POST. Para mayor información el proceso de conexiones HTTP entre el Sujeto y la TSA se explica en la sección 5.1, pasos 3 y 4.

Las dos clases principales de este módulo son:

TSAHttpServer: es una clase similar a la clase CERTILOCHttpServer, se trata de un servidor de HTTP cuya misión es recibir una petición de archivo y devolver dicho archivo. Con la salvedad de que en el caso de la TSA, está preparada para recibir un archivo TimeStampRequest.xml en los datos POST de la petición HTTP. Estos datos POST los vuelca a un fichero físico llamado TimeStampRequest.xml, y después invoca a el método CreateTSAResponse de la clase TSAManager.

TSAManager. El método CreateTSAResponse es el encargado de generar la respuesta de la TSA. Realiza las siguientes operaciones:

1. Instancia un objeto de la clase TimeStampRequest tomando el fichero físico TimeStampRequest.xml que generó la TSAHttpServer antes de invocar al TSAManager.
2. Genera un nuevo serialNumber tomando el último serial number emitido por la TSA incrementado en 1. Los serial numbers generados se guardan en un archivo de texto (parámetro de configuración TSA.SERIALNUMBERS.FILENAME)
3. Instancia un objeto de la clase TimeStampResponse tomando como datos de entrada las credenciales de la TSA (almacén de claves, nombre de la clave, etc, que son los parámetros de configuración cuyo nombre comienza por “KEYSTORE.TSA”), el nuevo serial number y el objeto de la clase TimeStampRequest.
4. El objeto TimeStampResponse se crea y junto con el todas sus entidades hijas (Status, TimeStampToken) teniendo en cuenta que la representación en xml de la hija TimeStampToken irá firmada con las credenciales de la TSA.

6.4.2 Diagrama de clases

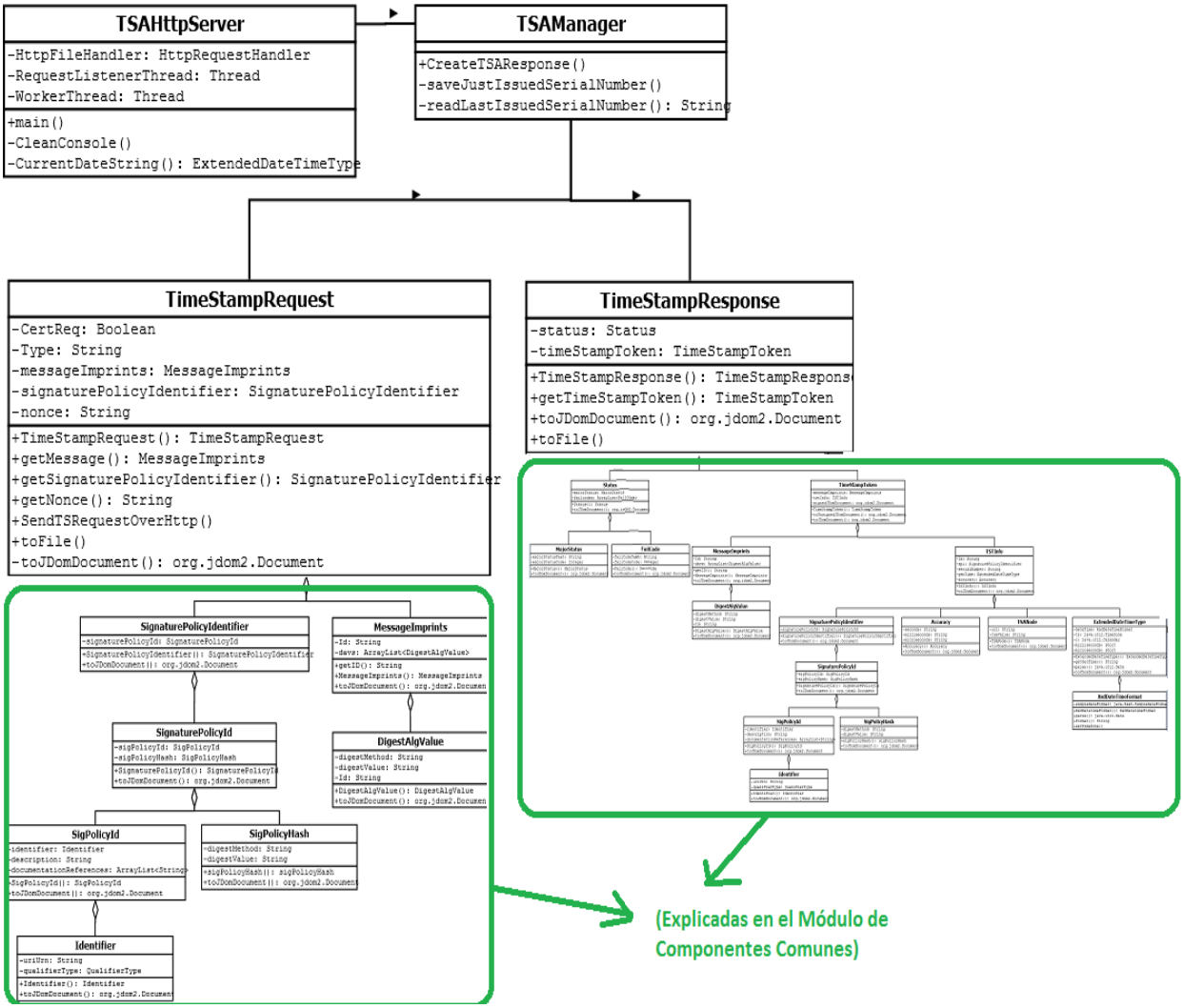


Figura 6.4.2 Diagrama de clases del módulo de la TSA

6.4.3 Dependencias de librerías externas

Este módulo depende de las librerías siguientes:

Nombre	Origen	Empaquetado	Propósito
Apache HttpComponents 4.1.4	http://hc.apache.org/	httpcore-4.1.4.jar	Hacer que CERTILOC sea un servidor http de Certificados Espacio Temporales
Joda Time 2.1	http://joda-time.sourceforge.net/	joda-time-2.1.jar	Permitir calcular diferencias entre fechas en la interfaz del Sellador Validador para mostrar la distancia recorrida y el tiempo entre las fechas del CET y de la TimeStampResponse
Jansi 1.9	http://jansi.fusesource.org/	jansi-1.9.jar	Para que la salida por consola sea con colores, se pueda borrar la pantalla, etc

6.4.4 Obtención de la fecha de la TSA

La TSA devuelve la información de tiempo en el nodo Gentime, que es un subnodo de TimeStampToken/TSTInfo. Como se puede apreciar en la siguiente figura:

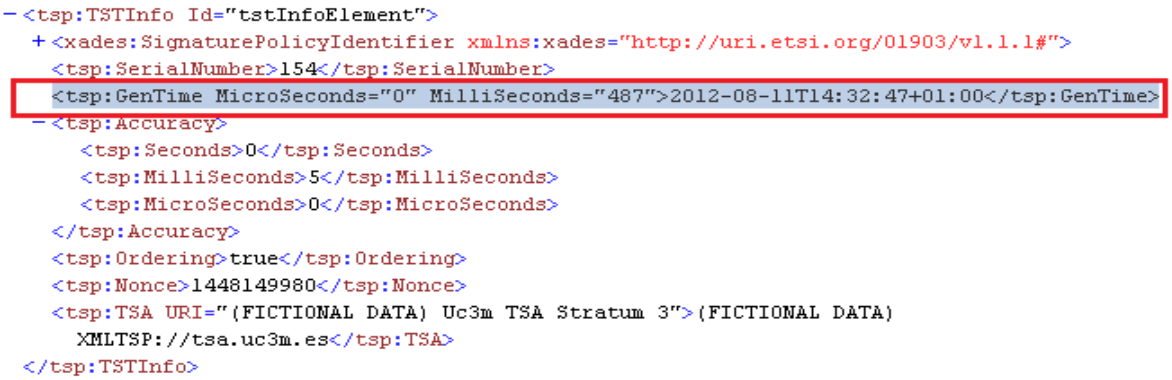


Figura: Situación de la información de tiempo de la TSA en el xml

La obtención de la fecha se realiza en el método constructor de la clase TSTInfo. Esta obtención está parametrizada de la siguiente manera:

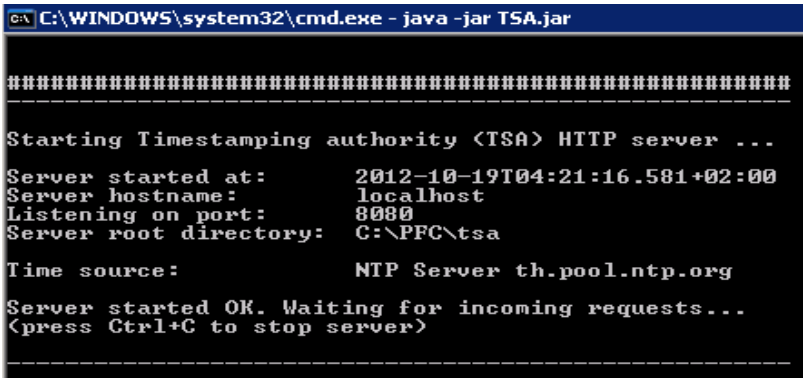
Hay tres modos de funcionamiento de la TSA en cuanto a la obtención de la fecha, el modo de funcionamiento estará marcado por varios parámetros de configuración y al arrancar la consola de la TSA se verá cual es en donde pone “Time source”

6.4.4.1 Obtención de la fecha mediante NTP

Para la obtención de la información de tiempo nos servimos de una librería que está disponible de manera gratuita y con licencia de código abierto de Apache. Dicha librería se llama org.apache.commons.net.ntp y hace uso del protocolo NTP versión 3.

En este caso se conectará al servidor NTP especificado en el parámetro TSA.TIME.DEFAULTNTPPOOL. Nosotros le hemos puesto el valor europe.pool.ntp.org que se trata de un “pool” de servidores NTP repartidos por europa. Este pool nos redirige al servidor NTP más apropiado según criterios de tráfico de red y cercanía geográfica. Para la información de tiempo el servidor nos devolverá Año, mes, día, hora, minutos, segundos y milisegundos.
Además tendremos que informar un resultado de precisión, expresado en el nodo Accuracy en segundos, milisegundos y microsegundos. El servidor NTP nos devuelve esta precisión en forma de milisegundos, por lo que solo informamos dicho valor, y en segundos y microsegundos ponemos el valor 0.

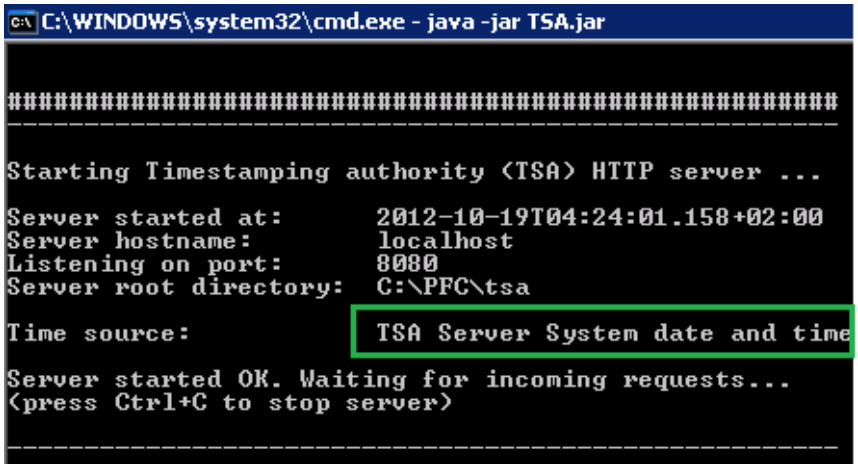
Este modo de funcionamiento se verá al arrancar la consola de la TSA, habrá un texto encabezado por “Time source” y a continuación el nombre del servidor NTP.



Es importante reseñar que la hora obtenida desde un servidor NTP viene con respecto a la zona horaria UTC, siendo la entidad cliente de NTP la encargada de transformar dicha hora a la hora local del sistema de instalación.

6.4.4.2 Obtención de la fecha mediante la fecha del sistema

En este caso se obtiene la fecha del sistema donde se ejecute la TSA. Para el valor de la precisión hemos escogido un valor arbitrario de 5 milisegundos. Este modo de funcionamiento se verá al arrancar la consola de la TSA:

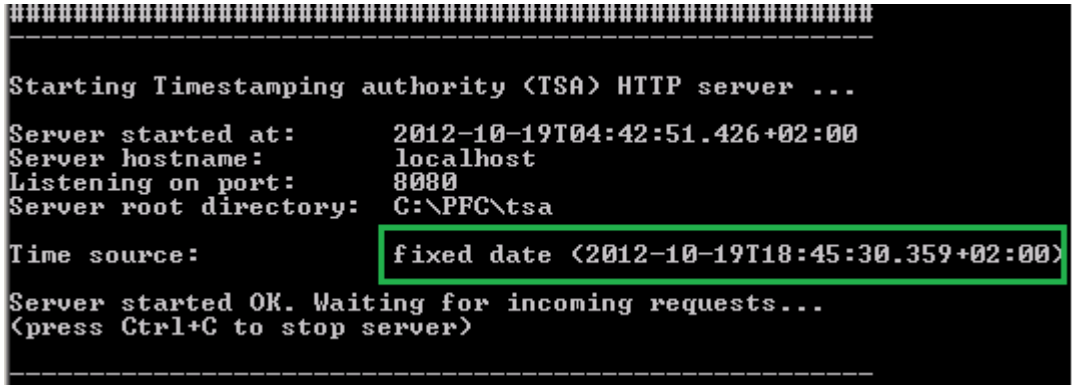


6.4.4.3 Forzar una determinada fecha

Este caso se da por el parámetro TSA.TIME.FORCEDATE. En este caso la fecha se tomará de una serie de parámetros de configuración, aquellos cuyo nombre comience por TSA.TIME.FORCEDATE :

```
TSA.TIME.FORCEDATE = true
TSA.TIME.FORCEDATE.DAY = 19
TSA.TIME.FORCEDATE.HOUR24H = 18
TSA.TIME.FORCEDATE.MICROSECOND = 0
TSA.TIME.FORCEDATE.MILLISECOND = 359
TSA.TIME.FORCEDATE.MINUTE = 45
TSA.TIME.FORCEDATE.MONTH = 10
TSA.TIME.FORCEDATE.SECOND = 30
TSA.TIME.FORCEDATE.TIMEZONE = GMT+02:00
TSA.TIME.FORCEDATE.YEAR = 2012
```

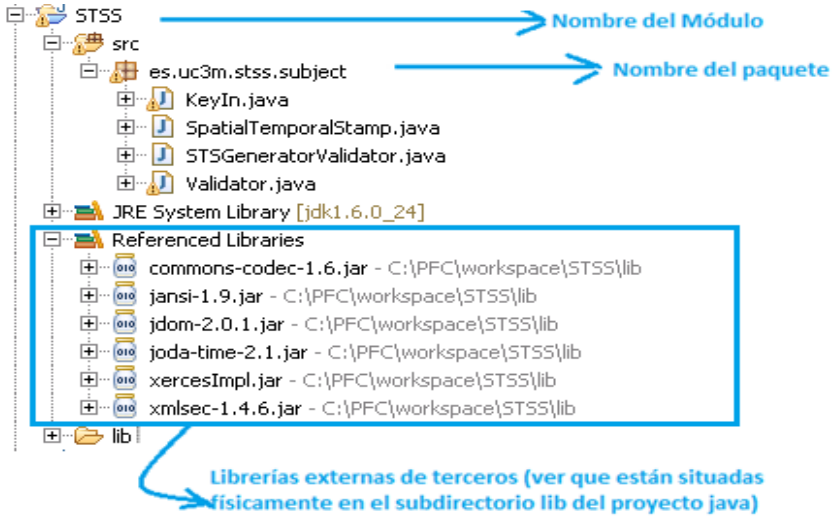
Para el valor de la precisión hemos escogido un valor arbitrario de 5 milisegundos. Este modo de funcionamiento se verá al arrancar la consola de la TSA:



6.5 Módulo Generador-Validador de SET

6.5.1 Visión general

Este módulo sirve para mostrar una interfaz gráfica de usuario representada por una serie de menus para que el usuario pueda iniciar la acción de generar el Sello Espacio Temporal, verificarlo y mostrarlo. El código de este módulo está implementado en un proyecto java llamado STSS dentro del workspace de código:



6.5.2 Diagrama de clases

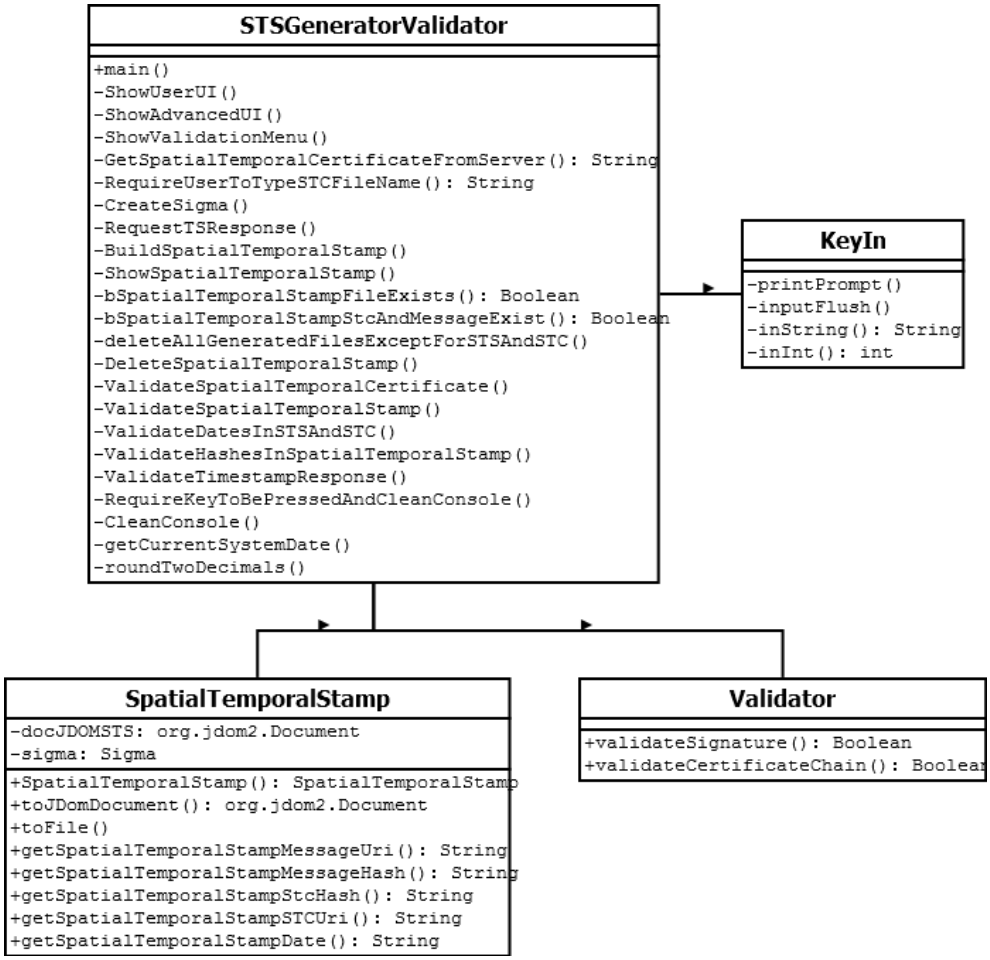


Figura 6.5.2 Diagrama de clases del módulo Generador/Validador

CLASE	PROPÓSITO	MOMENTOS EN QUE ESTÁ INSTANCIADA
STSGeneratorValidator	Esta clase arranca la consola principal del programa y maneja las distintas opciones de menú y la interacción con el usuario.	Todo momento en que esté ejecutando el programa
KeyIn	Esta clase sirve de ayuda para manejar el que el usuario pulse cada opción de menú.	Todo momento en que esté ejecutando el programa
SpatialTemporalStamp	Esta clase sirve para componer el Sello Espacio Temporal. En su constructor esta clase recibe como parámetros el archivo de Sigma y el archivo de TimeStampResponse. Con ellos compone un documento jdom similar al documento jdom de sigma, solo que el nodo raíz se llama SpatialTemporalStamp en lugar de "Sigma" y que dentro incluye el nodo TimeStampToken extraído del TimeStampResponse. El documento jdom resultante lo almacena en un miembro de la clase llamado docJDOMSTS, que después se pasa a archivo xml para dar con el STS final.	En la opción de menú 4 del menú avanzado o bien en la ejecución de la opción 1 del menú de usuario final después de recibir el TimeStampResponse de la TSA.
Validator	Esta clase sirve para validar firmas digitales en documentos xml y para validar rutas de certificación	Al ejecutar el menú validar.

6.5.3 Dependencias de librerías externas

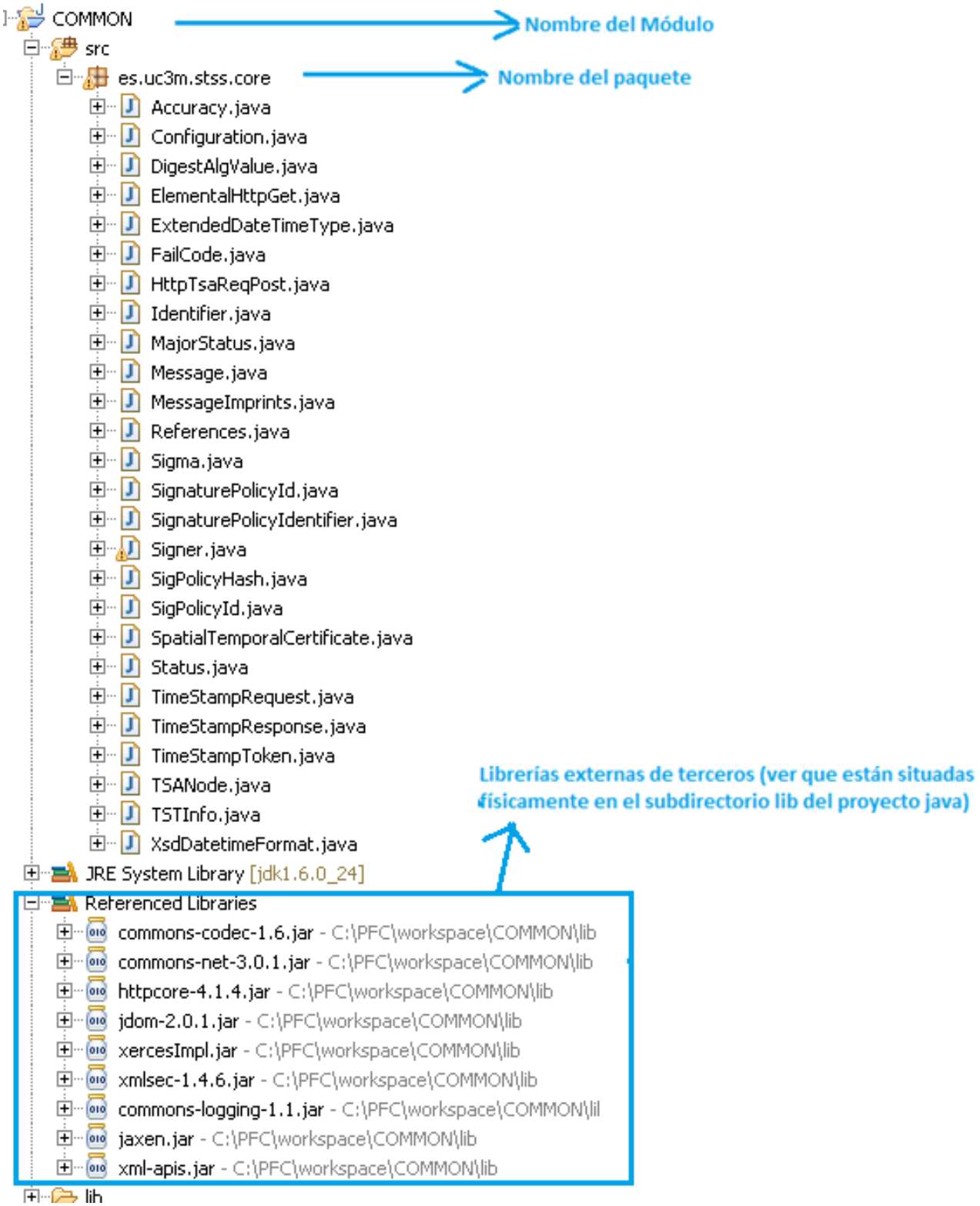
Este módulo depende de las librerías siguientes:

Nombre y versión	Origen	Empaquetado	Propósito
Apache Commons Codec 1.6	http://commons.apache.org/codec/	commons-codec-1.6.jar	Permitir transformar un resumen en formato array de bytes a hexadecimal
Jansi 1.9	http://jansi.fusesource.org/	jansi-1.9.jar	Para que la salida por consola sea con colores, se pueda borrar la pantalla, etc
JDOM2.0.1	http://www.jdom.org/	jdom-2.0.1.jar	Permitir el manejo en memoria de documentos xml conforme al modelo de objetos de documento (DOM) con comodidad
Joda Time 2.1	http://joda-time.sourceforge.net/	joda-time-2.1.jar	Permitir calcular diferencias entre fechas en la interfaz del Sellador Validador para mostrar la distancia recorrida y el tiempo entre las fechas del CET y de la TimeStampResponse
Apache Santuario 1.4.6	http://santuario.apache.org/	xmlsec-1.4.6.jar	Para manipular instancias de la clase XMLSignature poder firmar y verificar firmas
Xerces	http://www.jdom.org/	xercesImpl.jar	Librería necesaria para el funcionamiento de Jdom y otras invocaciones desde código donde se usa el DOM clásico. Viene incluida en el paquete de descarga de Jdom en la subcarpeta lib.

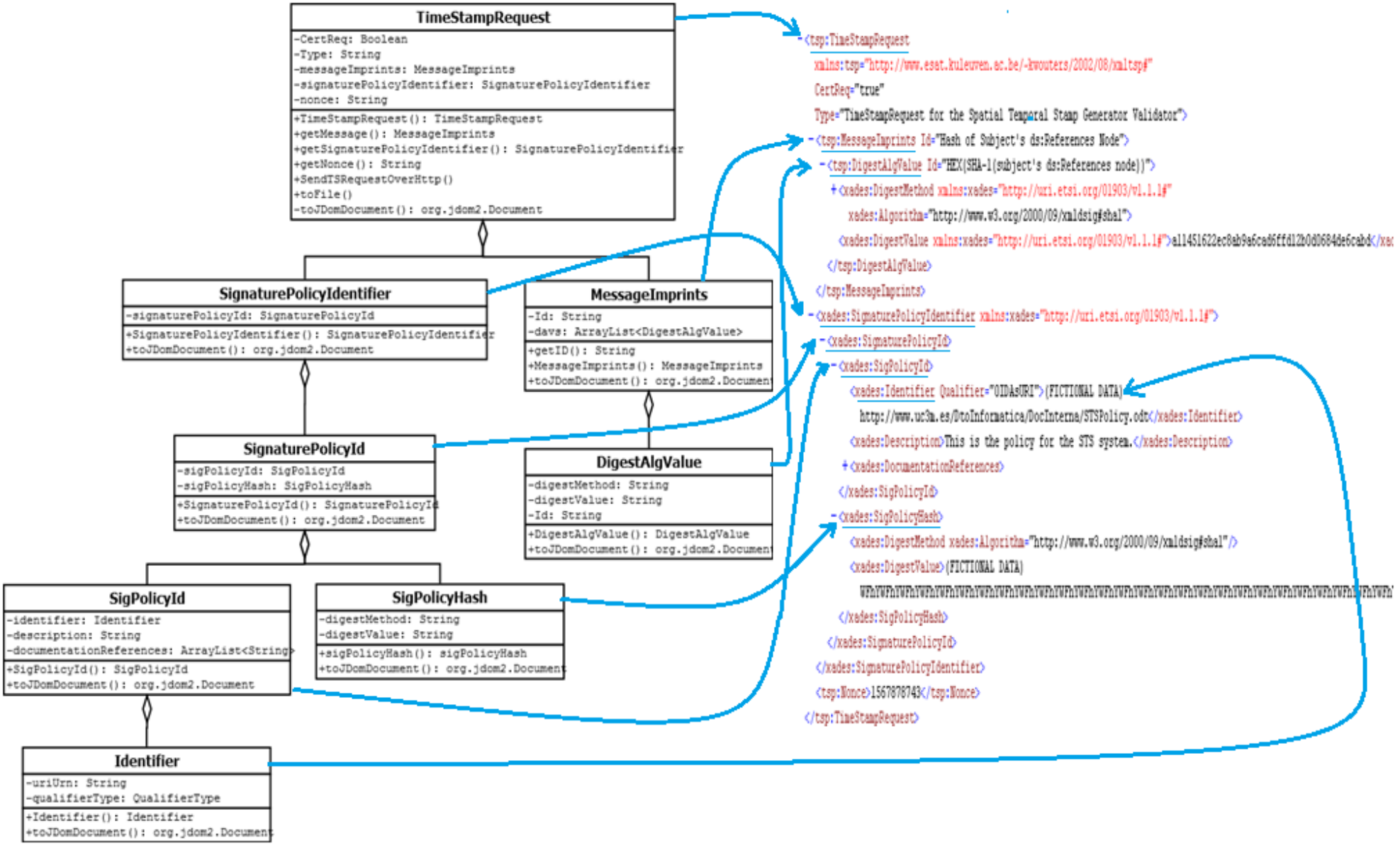
6.6 Módulo de componentes comunes

6.6.1 Visión general

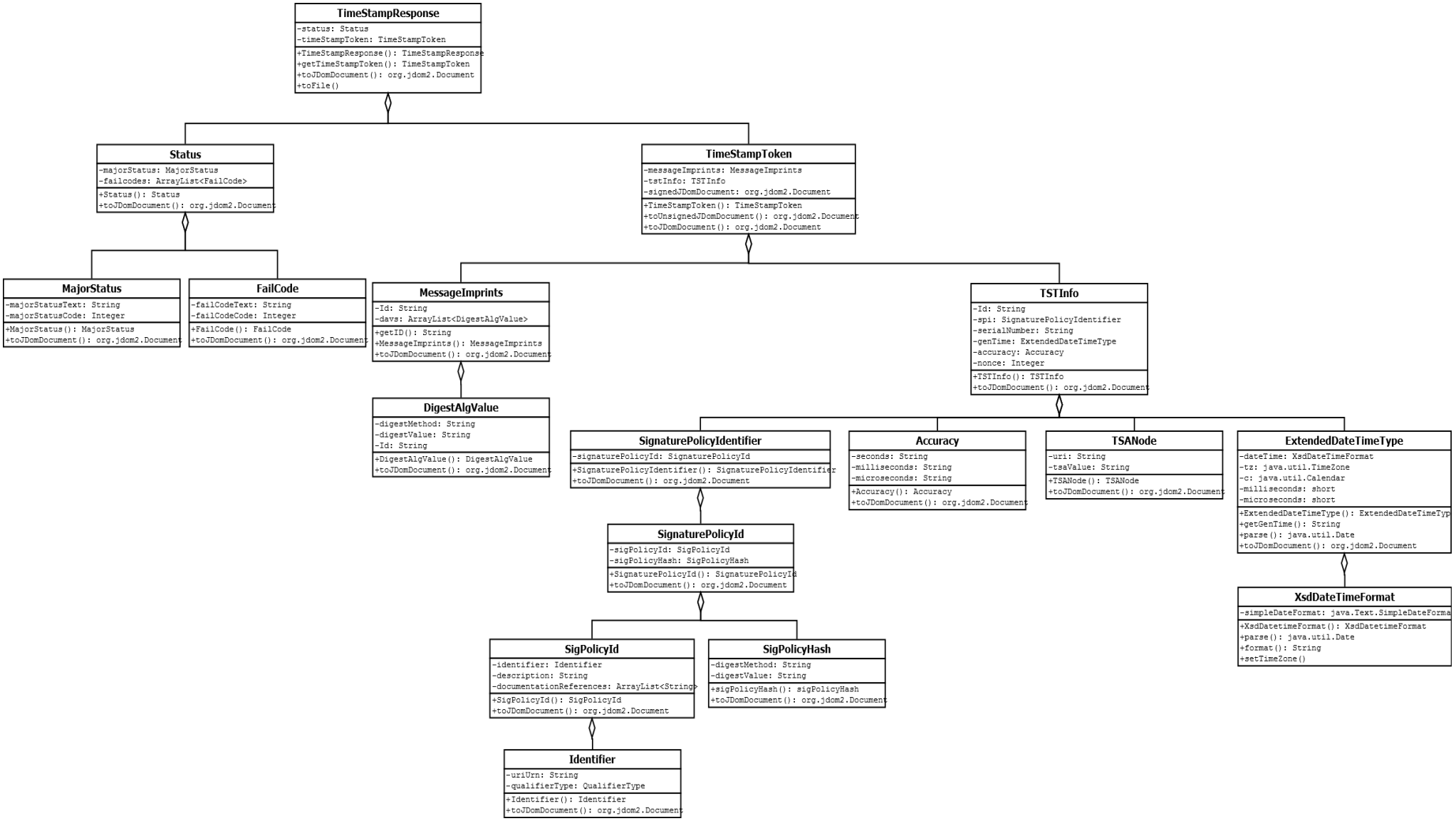
El código de este módulo está implementado en un proyecto java llamado COMMON dentro del workspace de código, como se muestra en la siguiente figura:



6.6.2 Diagrama de clases para la parte de TimeStamp Request



6.6.3 Diagrama de clases para la parte del TimeStamp Response




```
- <tsp:TimeStampResponse
   xmlns:tsp="http://www.esat.kuleuven.ac.be/~kwouters/2002/08/xmltsp#">
  - <tsp:Status>
      <tsp:MajorStatus Code="0">Issued TimeStamp OK</tsp:MajorStatus>
    </tsp:Status>
  - <tsp:TimeStampToken>
    - <tsp:MessageImprints Id="Hash of Subject's ds:References Node">
      - <tsp:DigestAlgValue Id="HEX(SHA-1(subject's ds:References node))">
        + <xades:DigestMethod xmlns:xades="http://uri.etsi.org/01903/v1.1.1#"
            xades:Algorithm="http://www.w3.org/2000/09/xmldsig#sha1">
          <xades:DigestValue xmlns:xades="http://uri.etsi.org/01903/v1.1.1#">a11451622ec8ab9a6cad6ffd12b0d0684de6cabd</xades:DigestValu
        </tsp:DigestAlgValue>
      </tsp:MessageImprints>
    - <tsp:TSTInfo Id="Time info from TSA">
      - <xades:SignaturePolicyIdentifier xmlns:xades="http://uri.etsi.org/01903/v1.1.1#">
        - <xades:SignaturePolicyId>
          + <xades:SigPolicyId>
          + <xades:SigPolicyHash>
          </xades:SignaturePolicyId>
        </xades:SignaturePolicyIdentifier>
        <tsp:SerialNumber>20</tsp:SerialNumber>
        <tsp:GenTime MicroSeconds="0" MilliSeconds="761">2012-08-27T13:01:49+01:00</tsp:GenTime>
      - <tsp:Accuracy>
          <tsp:Seconds>0</tsp:Seconds>
          <tsp:MilliSeconds>5</tsp:MilliSeconds>
          <tsp:MicroSeconds>0</tsp:MicroSeconds>
        </tsp:Accuracy>
        <tsp:Ordering>true</tsp:Ordering>
        <tsp:Nonce>1567878743</tsp:Nonce>
        <tsp:TSA URI="(FICTIONAL DATA) XMLTSP://tsa.uc3m.es">(FICTIONAL DATA) Uc3m TSA
          Stratum 3</tsp:TSA>
      </tsp:TSTInfo>
    + <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
        xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
        xmlns:xades="http://uri.etsi.org/01903/v1.1.1#"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        Id="TimeStampResponseSignatureElement">
    </tsp:TimeStampToken>
```

6.6.4 Dependencias de librerías externas

Nombre	Origen	Empaquetado	Propósito
Apache Commons Net 3.0.1	http://commons.apache.org/net/	commons-net-3.0.1.jar	Para acceder a servidores de NTP para obtener la fecha y hora de la TSA
Commons Logging 1.1	http://commons.apache.org/logging/	commons-logging-1.1.jar	Librería necesaria para la librería Apache Santuario, viene incluida en el paquete zip de descarga del mismo, en la subcarpeta lib.
Apache Commons Codec 1.6	http://commons.apache.org/codec/	commons-codec-1.6.jar	Permitir transformar un resumen en formato array de bytes a hexadecimal
Apache Santuario 1.4.6	http://santuario.apache.org/	xmlsec-1.4.6.jar	Para manipular instancias de la clase XMLSignature poder firmar y verificar firmas
Apache HttpComponents 4.1.4	http://hc.apache.org/	httpcore-4.1.4.jar	Hacer que CERTILOC sea un servidor http de Certificados Espacio Temporales
Joda Time 2.1	http://joda-time.sourceforge.net/	joda-time-2.1.jar	Permitir calcular diferencias entre fechas en la interfaz del Sellador Validador para mostrar la distancia recorrida y el tiempo entre las fechas del CET y de la TimeStampResponse
JDOM2.0.1	http://www.jdom.org/	jdom-2.0.1.jar	Permitir el manejo en memoria de documentos xml conforme al modelo de objetos de documento (DOM) facilmente
Xerces	http://www.jdom.org/	xercesImpl.jar	Librería necesaria para el funcionamiento de Jdom y otras invocaciones desde código donde se usa el DOM clásico. Viene incluida en el paquete de descarga de Jdom en la subcarpeta lib.
Jaxen 1.1.3	http://www.jdom.org/	jaxen-1.1.3.jar	Librería necesaria para el funcionamiento de Jdom y otras invocaciones desde código donde se usa el DOM clásico. Viene incluida en el paquete de descarga de Jdom en la subcarpeta lib.
Xml Apis	http://www.jdom.org/	xml-apis.jar	Librería necesaria para el funcionamiento de Jdom y otras invocaciones desde código donde se usa el DOM clásico. Viene incluida en el paquete de descarga de Jdom en la subcarpeta lib.

7. PLAN DE PRUEBAS

7.1 Introducción

Aparte de una serie de pruebas funcionales que detallaremos en las siguientes secciones, algunas pruebas son mediante software y las hemos realizado mediante una serie de servicios que son:

- Validación de firmas: Puesto que por un lado tenemos la firma que por parte del Sujeto se hace del Sigma y por otro lado la que hace la TSA de la TimeStampResponse, estas firmas necesitan ser validadas para verificar la integridad y no repudio del contenido firmado.
- Validación de rutas de certificación: Ahora que en las firmas ds:Signature incluímos los certificados de clave pública no solo del firmante sino de aquellas autoridades de certificación de las que depende, se hace necesario un mecanismo que permita validar que dichos certificados sean válidos.
- Validación de anterioridad de la fecha que figura en el Certificado Espacio Temporal con respecto a la fecha que devolvió la TSA.
- Validación de resúmenes: Puesto que ciertas partes del xml no se intercambian tal cual entre Sujeto y TSA, sino que se envían sus resúmenes, se hace necesario verificar la integridad de dichas informaciones resumidas, como por ejemplo la entidad Mensaje, que no se envía como tal sino su resumen en SHA-1.

7.2 Validación de firmas

La situación de partida es que en el software anterior teníamos un servicio de validación que recibía como parámetros el nombre del fichero xml a validar y una variable booleana que indica si estamos validando la firma de la tsa (la firma que hace del TimeStampResponse) o en caso contrario estamos validando la firma del sujeto (la firma que hace de Sigma). Este funcionamiento es rígido y poco intuitivo y debe ser mejorado.

En el presente proyecto las necesidades de validación han aumentado en dos aspectos:

- Se hace necesario un servicio de validación que no solo sirva para las firmas de la TSA y el sujeto, sino también para la firma que hace CERTILOC del Certificado Espacio Temporal.
- Por extensión, deseamos un servicio de validación que sirva para validar cualquier nodo ds:Signature de cualquier xml contenedor.

Por lo tanto el nuevo servicio de validación propuesto en la clase Validator, es un servicio tal que reciba como parámetros:

- el nombre del archivo xml que contiene la firma
- el identificador (Id) del nodo ds:Signature a verificar dentro de dicho archivo, asumiendo que puede haber varios nodos ds:Signature pero cada uno tiene un identificador único. El servicio buscará dicho nodo ds:Signature para validarlo.

```
public static boolean validate(String containerFile, String signatureNodeId)
```

7.3 Pruebas de rutas de certificación

Cada uno de los nodos ds:Signature objeto de este proyecto contiene un nodo KeyInfo que sirve para identificar al firmante. Este nodo KeyInfo puede contener 1 a N certificados pertenecientes a una cadena de certificación, como se explica en la sección 4.4.4 de la especificación de XML Signature. El primer certificado será el firmante, a continuación el certificado de su emisor, y así sucesivamente hasta completar la cadena de certificación en el certificado raíz.

Como explicamos en la sección "Estructura de Certificación final", esta estructura se va aplicar a las diversas firmas implicadas en el proceso de creación de un Sello Espacio Temporal.
El objetivo es definir un servicio de validación que sea utilizable para cualquier nodo ds:signature de cualquier xml. El servicio que hemos definido recibe como parámetros el nombre del xml que contiene la firma y una ruta en formato XPath para encontrar el nodo ds:Signature.

```
public static boolean validateCertificateChain(String containerFile, String sxPathExpressionForDsSignatureNode)
```

El uso de Xpath para encontrar el nodo ds:Signature se ha hecho por motivos de eficiencia ya que la cadena XPath nos permite encontrar el nodo deseado de manera directa.

Ejemplo de uso:

Si quisiéramos validar la firma de Sigma.xml (esta es por parte del Sujeto) tendríamos que invocar el servicio de la siguiente forma:

```
- <tsp:Sigma xmlns:tsp="http://www.esat.kuleuven.ac.be/~kwouters/2002/08/xmltsp#">
  + <tsp:File tsp:Id="Message" tsp:URI="loren_ipsum.txt">
  + <tsp:Certificate tsp:Id="Certiloc STC" tsp:URI="signedSTC.xml">
  + <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
      xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
      xmlns:xades="http://uri.etsi.org/01903/v1.1.1#"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      Id="SigmaSignatureElement">
  </tsp:Sigma>
```

Validator.validateCertificateChain("Sigma.xml","tsp:Sigma/ds:Signature");

Figura: Ejemplo de validación de cadena de certificación del fichero Sigma.xml

Cómo se hace la validación (descripción a alto nivel):

1. Obtener el nodo ds:Signature mediante la expresión XPath
2. Obtener una lista de elementos X509Certificate hijos del subnodo X509Data de ds:signature
3. Tomar el último elemento de dicha lista (será el certificado raíz) y ponerlo como Trust Anchor de la cadena de certificación.
4. Utilizar un objeto de la clase java-security.cert.CertPathValidator, su método .validate validará la ruta de certificación tomando la lista obtenida en el paso 2 y el Trust Anchor obtenido en el paso 3 como parámetros.

Prueba de caducidad de certificados: Se pone como fecha de sistema una fecha posterior a la fecha de validez de los mismos y efectivamente el cert path dice que no es válido

7.4 Validación de anterioridad de la fecha del Certificado Espacio Temporal

El objetivo de esta validación es comprobar que la firma Sigma se generó después del instante inicial en el que se generó el Certificado Espacio Temporal y anteriormente al Sello de anterioridad emitido por la TSA.
Para obtener el instante inicial, que es el instante del Certificado Espacio Temporal emitido por CERTILOC, nos vamos al nodo saml:Attribute cuyo friendlyName es "TimeInfo". Va firmado por CERTILOC:

Figura: Instante inicial o "Sello de posterioridad"

Como puede observarse la fecha es 6 de agosto de 2012 a las 17:28, esto significa que el Sujeto se encontraba en la localización certificada por certiloc en dicha fecha.

Para obtener el instante de anterioridad contenido en el Sello de Anterioridad emitido por la TSA, nos vamos al nodo GenTime del TimeStampToken devuelto firmado por la TSA:

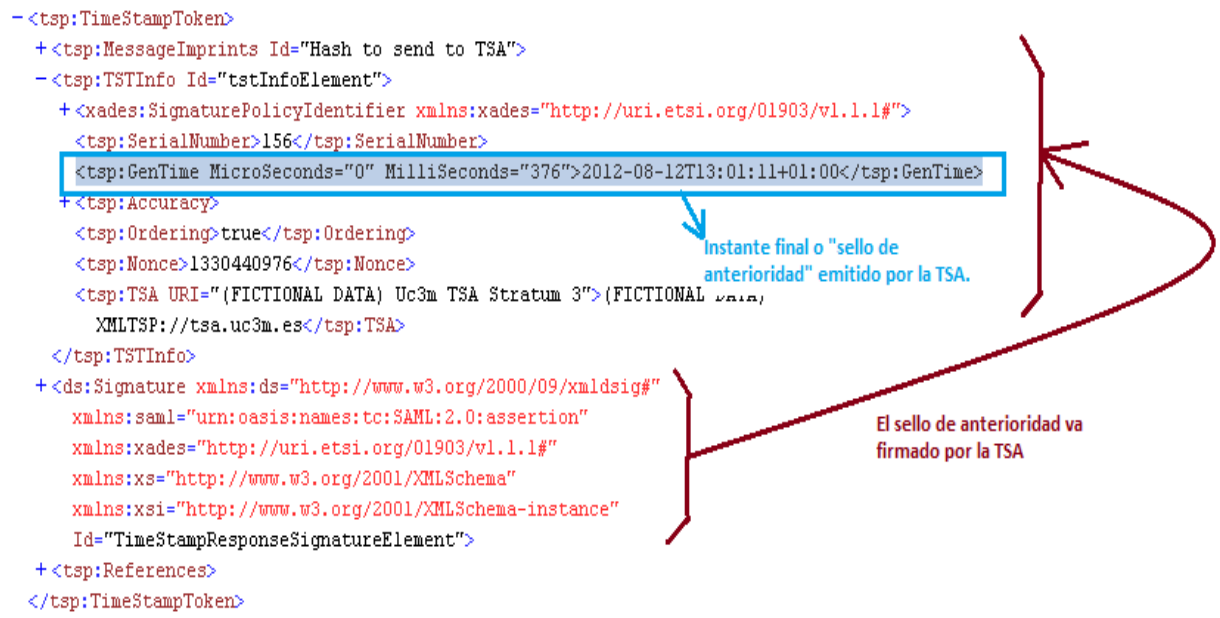


Figura: Instante final o "sello de anterioridad"

Como puede observarse la fecha es 12 de agosto de 2012 a las 13:01.
La verificación de posterioridad consiste en que verificamos que la fecha firmada por la TSA es posterior a la fecha firmada por CERTILOC. De esta forma nos aseguramos lo siguiente:
El sujeto S el 12 de agosto estaba en posesión del mensaje, después de haber estado en la localización inicial el 6 de agosto.
Es decir que el mensaje ha estado en poder del sujeto en algún momento entre el 6 de agosto cuando estuvo en la localización inicial y el día 12 de agosto

El resultado de la validación de anterioridad se muestra por consola durante las validaciones del sello espacio temporal, como se ve en la figura siguiente:

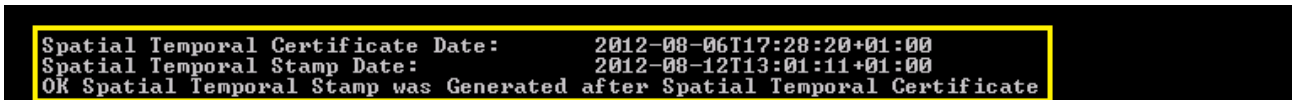


Figura: Resultado de validación de anterioridad por consola

7.5 Comprobación del resumen del Mensaje

Creación de resumen del fichero de Mensaje (va al nodo File del Sello Espacio Temporal):

Como se puede apreciar en la figura, al instanciar un objecto de la clase Message le pasamos el nombre del fichero que lo contiene, primero leemos todo el fichero de mensaje en un array de bytes, instanciamos la clase MessageDigest para SHA-1 y el resumen resultante lo ponemos en formato hexadecimal:

```
public Message(String sMessageFile) throws Exception
{
    Namespace TspNs = Namespace.getNamespace(Configuration
            .getInstance().getProperty("NS.TSP.PREFIX"), Configuration
            .getInstance().getProperty("NS.TSP.URI"));

    //Compute hash of message file
    String sDigestAlgorithm = MessageDigestAlgorithm.ALGO_ID_DIGEST_SHA1;

    this.sMessageHash = computeHash(sMessageFile);

    hashOfMessage = new DigestAndValue(
            sDigestAlgorithm,
            sMessageHash,
            "HEX(SHA-1(Message file))",
            Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DIGI
            TspNs);

    Id = Configuration.getInstance().getProperty("TIMESTAMPREQUEST.DEFAUL

    Uri =   sMessageFile;

}
```

```
private String computeHash(String sMessageFile)throws Ex
{
    String sDocHash = null;

    // Read message file content
    FileInputStream fis = null;
    byte[] dataInByteArray = null;
    fis = new FileInputStream(sMessageFile);
    int bytes = fis.available();
    dataInByteArray = new byte[bytes];
    fis.read(dataInByteArray);
    fis.close();

    //Compute hash of message file
    MessageDigest md;
    md = MessageDigest.getInstance("SHA-1");
    md.update(dataInByteArray);
    byte[] digest = md.digest();
    sDocHash = Hex.encodeHexString(digest);

    return sDocHash;
}
```

Resultado en xml:

Como puede apreciarse, tenemos el valor resumen en formato hexadecimal.

```
-<tsp:File tsp:Id="Message" tsp:URI="loren_ipsum.txt">
  -<tsp:DigestAlgValue Id="HEX(SHA-1(Message file))">
    -<xades:DigestMethod xmlns:xades="http://uri.etsi.org/01903/v1.1.1#"
       xades:Algorithm="http://www.w3.org/2000/09/xmldsig#sha1">
     </xades:DigestMethod>
     <xades:DigestValue xmlns:xades="http://uri.etsi.org/01903/v1.1.1#">42c2f1660e30ff825e4fb70bcb317ee7cb554976</xades:DigestValue>
   </tsp:DigestAlgValue>
 </tsp:File>
```

Comprobación:

Tomamos el valor resumen del nodo digestValue y esto lo comparamos con el resumen que se calcula al instanciar el objeto de la clase Message de nuevo

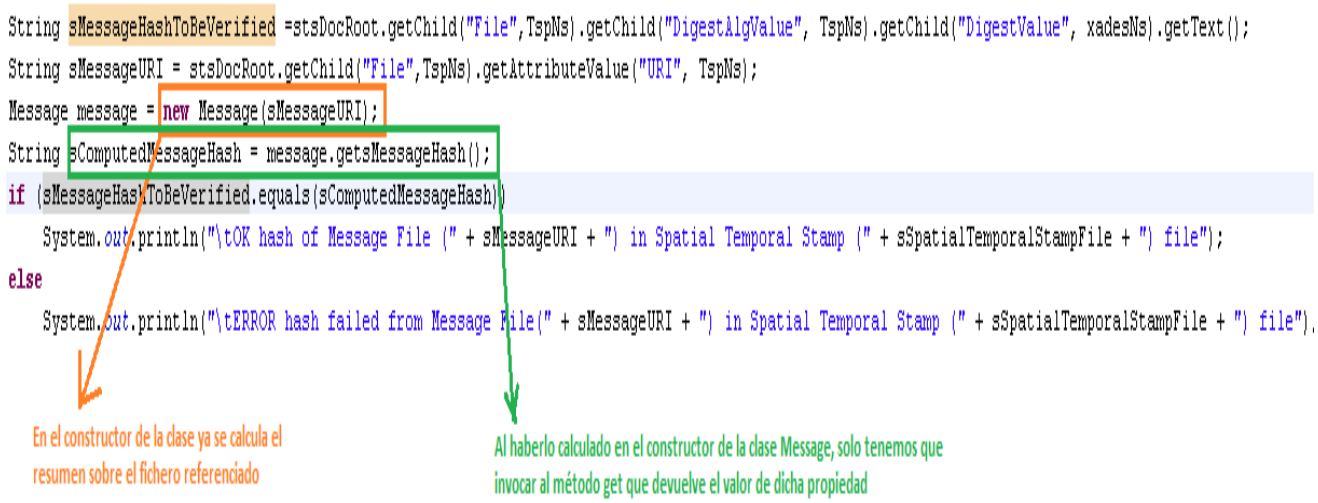


Figura 7.5 Comprobación del resumen del mensaje

7.6 Pruebas de petición y respuesta HTTP

El objeto de estas pruebas es verificar que no hay ningún problema en las operaciones de petición HTTP por parte del sujeto. Para hacer las pruebas utilizamos un plugin para el navegador Mozilla Firefox llamado HttpRequester, que nos permite efectuar una petición HTTP personalizada con los parámetros y cabeceras deseados y ver el resultado que nos devuelve el servidor.

7.6.1 Petición / respuesta HTTP de Certificado Espacio Temporal a CERTILOC por parte del Sujeto

Como dijimos en 4.2, el formato de la petición es el siguiente:
GET HTTP://[NOMBRE SERVIDOR CERTILOC]:[NºPUERTO]/[RUTA/AL/ARCHIVO]/[NOMBREARCHIVO]
Para hacer la prueba en el servidor local ponemos como url http://localhost:9090/signedSTC.xml y como nos muestra la figura 3 tenemos una respuesta 200 OK:

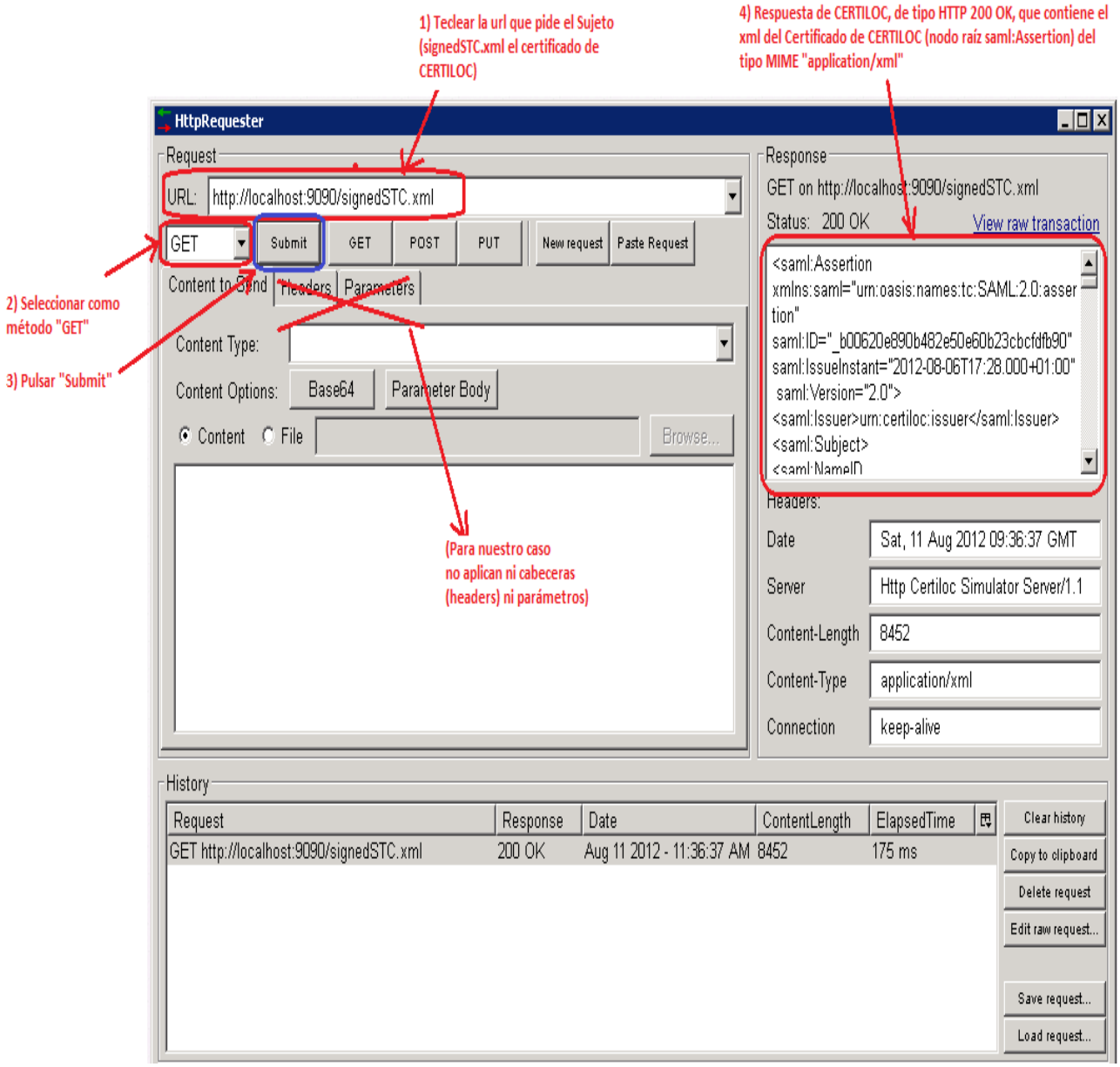


Figura 3: Prueba de petición y respuesta a CERTILOC

7.6.2 Petición / respuesta HTTP de TimeStamp Token a la TSA por parte del Sujeto

Como dijimos en 4.2, el formato de la petición es el siguiente:
POST HTTP://[NOMBRE SERVIDOR TSA]:[NºPUERTO]/[RUTA/AL/ARCHIVO]/TimeStampResponse.xml

Para hacer la prueba deseada en el servidor local ponemos como url http://localhost:8080/TimeStampResponse.xml, seleccionamos como tipo POST y en la parte del contenido ponemos que es del tipo application/xml y en la caja de texto del contenido ponemos el contenido de una petición TimeStampRequest.xml (previamente tenemos que haberla generado)

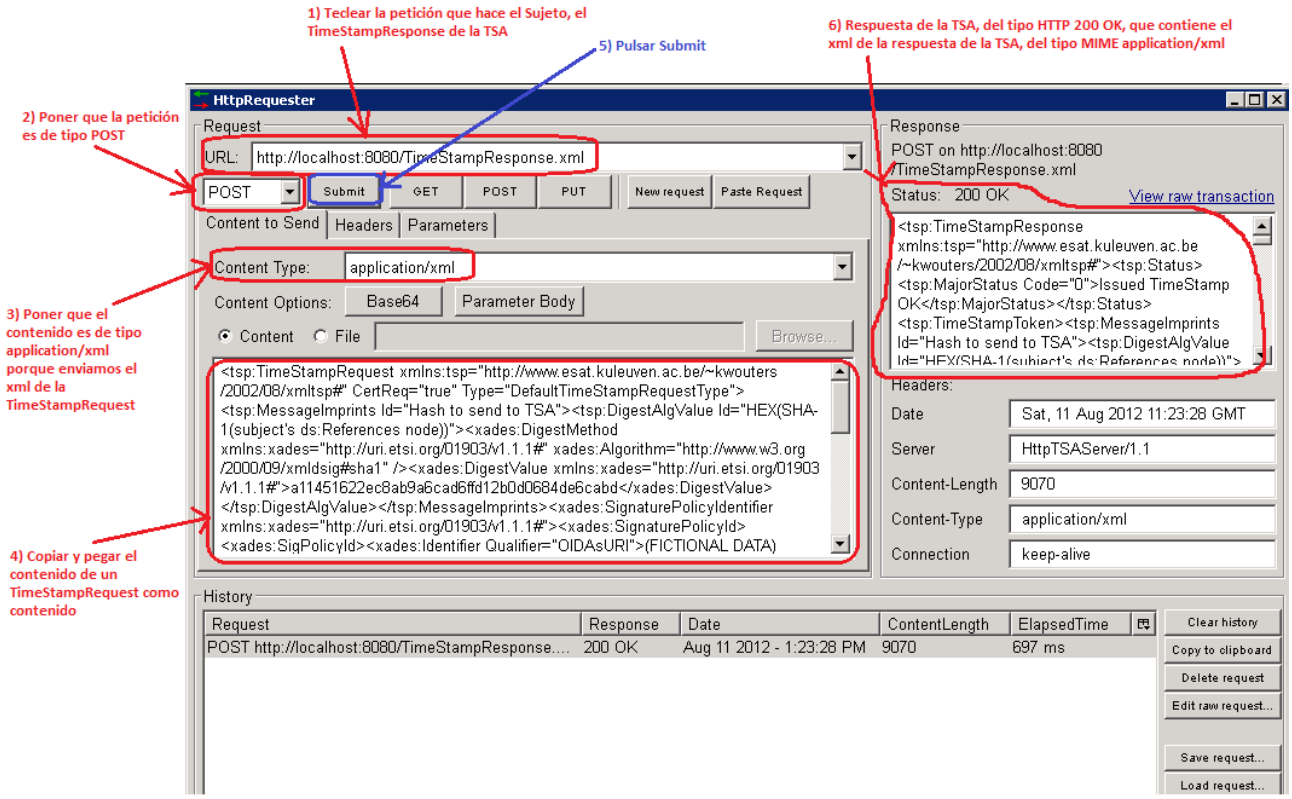


Figura 4: Prueba de petición y respuesta a la TSA

7.6.3 Acerca del Keep-alive

Algo digno de reseñar es que para los servidores HTTP tanto de la TSA como de CERTILOC se realiza un Keep-alive, que es un tipo de conexión Http que sirve para no cerrar la conexión TCP y así evitar abrir y cerrar una conexión TCP para cada uno de los recursos de una página html compleja en el ámbito de Internet. Hemos decidido mantener esta funcionalidad porque nos ayuda a optimizar el rendimiento del servidor Http. Por otro lado, el plugin HttpRequester que utilizamos para las pruebas en la sección 7.6 no utiliza esta conexión Keep-alive (aunque el servidor en la respuesta Http le indica que el Keep-alive está disponible) sino que cierra la conexión al terminar. Por ello en la consola de la TSA y de CERTILOC sale el mensaje "i/o error timed out" al utilizar dicho plugin, pero la TimeStampResponse se ha servido correctamente y la TSA sigue abierta a nuevas conexiones:

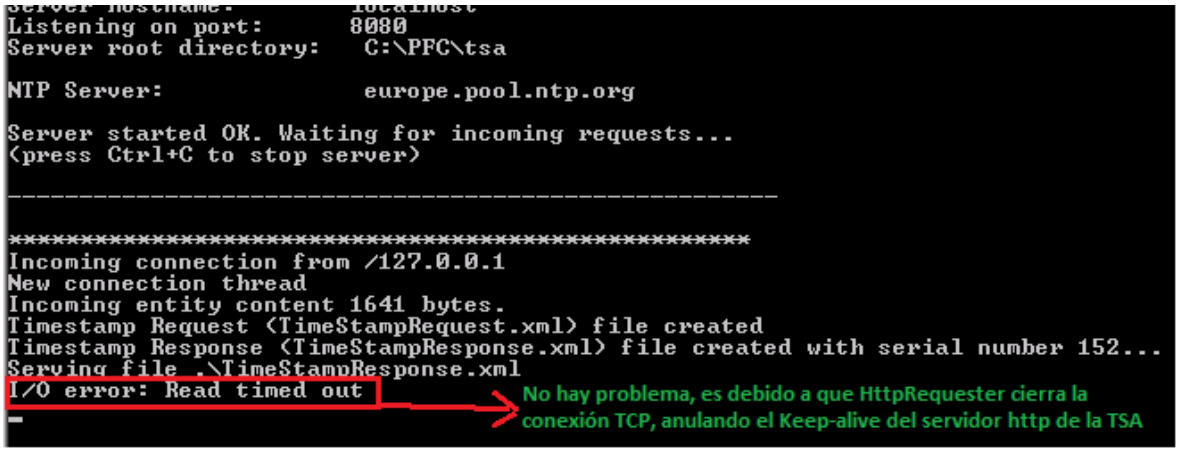


Figura 5: Explicación del error en consola al utilizar HttpRequester

7.7 Pruebas de esquema

Se ha comprobado que todos los documentos xml generados son válidos respecto al esquema definido para el Protocolo de Sellado Espacio Temporal (PSET).

7.8 Pruebas de tamaño del fichero Mensaje

La aplicación permite que el usuario seleccione el archivo de Mensaje deseado. Este archivo puede tener cualquier formato pues se admite todo tipo de archivos sean binarios o de texto. Sin embargo, el contenido del archivo se carga en la memoria del equipo durante el procesamiento del archivo de Mensaje, lo cual tiene implicación de uso de recursos hardware del equipo a nivel de memoria de acceso aleatorio (RAM) y procesador. El tamaño de dicho archivo estará limitado por la máquina virtual java que a su vez depende de los recursos hw/sw del sistema. El programa está preparado para abortar su funcionamiento cuando este archivo exceda el límite de tamaño .

Se han realizado diversas pruebas con tamaños de archivo de Mensaje, y se ha establecido (en uno de los equipos de desarrollo) que dicho límite oscila entre 168Mb y 180Mb.

7.9 Pruebas de ejecución en otro sistema operativo

Se ha probado a ejecutar la aplicación en otro sistema operativo, siendo el elegido Ubuntu Linux. El motivo de elegir este sistema operativo es que la máquina del Departamento de Informática que corre el demostrador de CERTILOC lo hace sobre ese sistema operativo. Las pruebas han sido igualmente satisfactorias.

8. CONCLUSIONES Y LÍNEAS FUTURAS

Este capítulo final del PFC presenta las conclusiones obtenidas por el autor así como las propuestas para futuras ampliaciones y/o mejoras del trabajo realizado.

8.1 Conclusiones

En este proyecto se han propuesto mejoras a la implementación del Protocolo Gonzalez-Tablas del proyecto para proveer servicios de Sellado Espacio-Temporal.

Se ha seguido un proceso de desarrollo mediante el Ciclo de Vida de Proyecto Software clásico, que separa las tareas a realizar en análisis, diseño, implementación y pruebas. Seguir estos pasos separados pero dependientes nos ha permitido optimizar al máximo el tiempo asignado a la realización de este proyecto.

Se han analizado las diferentes entidades implicadas en el proceso de Sellado Espacio Temporal, que son el Sujeto (cliente), la TSA (servicio de sellado temporal, o "SST") y CERTILOC (servicio de acreditación/certificación espacio-temporal, o "SAET") y se ha identificado el grado de cumplimiento del software implementado en el proyecto con respecto a estas tres áreas funcionales, obteniendo una lista de requisitos a implementar en el presente proyecto. Principalmente se ha identificado que el proyecto anterior no está implementado estrictamente como un servicio (al ejecutarse todo dentro de la misma máquina) y el aspecto de la interfaz de usuario es mejorable, entre otros aspectos que se detallan en la sección 4. ANÁLISIS DEL PROBLEMA Y ALTERNATIVAS DE SOLUCIÓN.

Una de las conclusiones que se extraen de este proyecto es la importancia de la modularización, puesto que nos ha ayudado decisivamente en la definición de este proyecto. Para contribuir a la modularización del software implementado, (un módulo para cada una de las áreas funcionales que acabamos de citar: Sujeto,TSA y CERTILOC) contamos con un módulo común que nos permite codificar aquellas entidades que son comunes a cada una de esas tres áreas funcionales, como por ejemplo la petición de sellado temporal, que está representada en una clase TimeStampRequest de la que hacen uso tanto el módulo de Sujeto como el módulo de la TSA). Esto ahorra tiempo de desarrollo y hace el código mas sencillo, legible, mantenible y poco redundante.

Otro de los aspectos positivos de importancia que hemos identificado es la configurabilidad de cada área funcional, lo cual redunda en un mantenimiento sencillo y poco invasivo, ya que hemos utilizado una serie de archivos de texto (ver la sección Apéndice C: Mantenimiento – propiedades configurables) con una serie de propiedades en formato clave-valor que nos permiten cambiar el comportamiento de cada módulo sin necesidad de depender de tareas de desarrollo software + compilación adicionales. De esta manera una persona con pocos conocimientos técnicos y que comprenda en qué consisten estos servicios puede razonablemente modificar el comportamiento de este software sin necesidad de tener conocimientos de programación de software.

Al haber analizado y diseñado este proyecto por áreas funcionales, para cada área funcional podemos extraer las siguientes conclusiones:

- Para el caso del cliente, que es el software principal, se ha identificado la necesidad de implementar una interfaz para el usuario y en la fase de análisis del proyecto se ha definido la estructura de dicha interfaz, implementándose la misma de manera satisfactoria con una interfaz a manejar por teclado con un mínimo de pulsaciones , un menú de usuario sencillo e incluso la posibilidad de visualizar la interfaz gráfica en dos niveles de complejidad configurables a voluntad.

- Para el caso de CERTILOC, puesto que existe ya un software para el mismo y la complejidad de utilizarlo se excede del ámbito del presente Proyecto, se ha implementado un simulador del mismo que nos permite visualizar esta parte del proceso de manera transparente, con la posibilidad de una utilidad de firmado que nos permite firmar un certificado espacio Temporal con las credenciales que hemos definido para este CERTILOC.

- Para el caso de la TSA, se ha implementado un servidor que supone una mejora sustancial respecto a la situación anterior, puesto que este nuevo servidor se puede ejecutar en una máquina independiente, es perfectamente configurable y permite escoger entre diversas opciones para obtener la información temporal.

Se ha mejorado también el aspecto técnico de las credenciales de firma que permiten a cada entidad principal (Sujeto, TSA, CERTILOC) acreditar las evidencias correspondientes en cada paso del proceso de Sellado Espacio Temporal. En este sentido, se ha definido una estructura de certificación con una serie de Autoridades de Certificación ordenadas de una manera jerárquica. El objeto de esta mejora es asemejar las credenciales de estas entidades a una situación real en la que los diversos servicios (SST,SASET) se sitúan de manera departamental ya sea en una empresa o en una organización gubernamental. Una vez más estas credenciales son configurables bajo ciertas restricciones como se explica detalladamente en la sección Apéndice C Mantenimiento - Mantenimiento de la estructura de certificación.

Por último, debemos mencionar que además de la multitud de mejoras a nivel tecnológico, otro de los énfasis que se han hecho es en mejorar la estructura de ciertas partes de los documentos xml implicados en el proceso, estas modificaciones han contribuido a aumentar la semántica contenida en dichos documentos como se explica detalladamente en la sección 5.4 Falta de Semántica de los documentos generados.

8.2 Líneas futuras

8.2.1 Necesidades funcionales

- **Devolución de estado de la TSA.**

  La TSA siempre retorna un Status de "sello emitido correctamente", y cuando hay algún problema o error de ejecución no devuelve nada. Esto es mejorable en el sentido de que se puede usar dicho nodo Status cuando haya algún problema con el tratamiento de la petición.

  El subnodo Status de la respuesta de la TSA (TimeStampResponse) ha sido establecido para retornar un código de éxito (Issued TimeStamp OK). Sería necesario modificar el módulo de la TSA para que si hay algún problema en el tratamiento de la petición, este problema se manifieste en un código o una serie de códigos en dicho nodo Status indicando el motivo del problema.

- **Implementación de una interfaz gráfica para un terminal Android.**

  Sería recomendable pues esto permitiría aprovechar las características gráficas de Android para hacer interfaces sencillas y con potencial de todo tipo de controles gráficos, incluyendo la posibilidad de, al mostrar un sello espacio temporal, mostrar una ventana de google maps donde se vean los posibles radios de distancia según la velocidad entre los instantes inicial y final del Sello Espacio Temporal. Durante el transcurso de este proyecto se estudió brevemente la posibilidad de realizar esta implementación y se propusieron algunos prototipos de pantallas de interfaz gráfica, desgraciadamente aunque al igual que en nuestro proyecto el lenguaje de programación de Android mayoritario es java, la compatibilidad de librerías java de tratamiento de documentos xml y criptografía no está todavía suficientemente soportada por dicho sistema operativo, que no dispone de una máquina virtual java equivalente a la instalada en sistemas de escritorio.

8.2.2 Necesidades de implementación

- **Uso completo de Xpath.**

  El método validate de la clase Validator, que se usa para validar una firma XMLSignature, se ha implementado recibiendo como parámetro de búsqueda el identificador (Id) del nodo ds:Signature a buscar. Habría sido más eficiente recibir como parámetro una expresión XPath de JDOM al igual que se hace en el método validateCertificateChain, sin embargo esto no ha sido posible porque la clase XMLSignature de la librería Apache Santuario no soporta el uso de JDOM para instanciar XML en memoria, en su lugar solo se puede instanciar la clase XMLSignature mediante la clase org.w3c.dom.document. Sería recomendable encontrar una manera de instanciar XMLSignature pudiendo utilizar XPath de JDOM. En diversos foros se ha expresado la necesidad de que XMLSignature soporte uso de JDOM, pero de momento no se ha encontrado nada concluyente.

- **Uso de OSCP.**

  Actualmente no se contempla el protocolo OSCP (Online Certificate Status Protocol) para la comprobación de la validez de los certificados, pero sería recomendable extender la comprobación de validez de certificados en ese sentido.

- **"Logging" de acciones**

  Mejora de la información al usuario por pantalla y registro (log) de las acciones ejecutadas en archivos formateados para poder hacer un seguimiento y estadística del uso del software de este proyecto. Para este fin se pueden usar librerías java como por ejemplo la popular *log4j*.

BIBLIOGRAFÍA

Tesis Doctorales y Proyectos de Fin de Carrera

[1] Álvaro Gascón y Marín de la Puente. Implementación de protocolos de sellado temporal y sellado espacio-temporal en XML para CERTILOC. Proyecto de Fin de Carrera. Universidad Carlos III de Madrid. 2008.

[2] Ana Isabel González-Tablas Ferreres. Arquitectura y mecanismos para la provisión de servicios de acreditación y sellado espacio-temporal. PhD thesis. Universidad Carlos III de Madrid. 2005.

[3] J.M de Fuentes García-Romero de Tejada. Diseño e implementación del sistema de localización de dispositivos móviles con conectividad limitada dotados de receptor gps para certiloc. Proyecto de Fin de Carrera. Universidad Carlos III de Madrid, 2007.

[4] José Carlos Calvo Martínez. Diseño e Implementación de la plataforma base de CERTILOC y del servicio de certificación espacio-temporal. Proyecto de Fin de Carrera. Universidad Carlos III de Madrid, 2007.

[5] John Páter Martínez de Leiva. Diseño e Implementación del Sistema de Políticas de Privacidad para el proyecto CERTILOC. Proyecto de Fin de Carrera. Universidad Carlos III de Madrid, 2009.

Congresos o reuniones

[6] Karel Wouters , B. Preneel, Ana Isabel González-Tablas Ferreres and Arturo Ribagorda. Towards an XML Format for Time-Stamps. In Proceedings of the ACM Workshop on XML Security, 2002.

[7] Ana Isabel González-Tablas Ferreres, Benjamín Ramos and Arturo Ribagorda. Protocolos de sellado espacio-temporal: Mejorando su precisión y disminuyendo el nivel de confianza requerido. Actas del Tercer Congreso Iberoamericano de Seguridad Informática (CIBSI'05). 2005

Normas y leyes

[8] W3C. XML-Signature syntax and processing. W3C Recommendation, 10 June 2008.
http://www.w3.org/TR/xmldsig-core/

[9] IETF. RFC 3161 (status: standards track). Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP). August 2001.
http://www.ietf.org/rfc/rfc3161.txt

[10] ETSI. XML Advanced Electronic Signatures (XAdES)
http://uri.etsi.org/01903/v1.4.1/

[11] IETF. RFC 1305 (status: draft standard, obsoleted by RFC 5905 NTPv4). Network Time Protocol (version 3). Specification, Implementation and Analysis.
http://datatracker.ietf.org/doc/rfc1305/

[12] Gobierno de España. Ley Orgánica 15/1999 de Protección de Datos de Carácter Personal.

[13] ISO. ISO 14516. Information Security – Security techniques – Guidelines for the use and management of Trusted Third Party Services.

Libros

[14]Elisabeth Castro. XML Guía de Aprendizaje. Editorial Prentice-Hall, 2001.

[15] Prince Sodhi. IT Project Management Handbook. Management Concepts, 2001.

Páginas o documentos electrónicos en la red

[16]W3Schools.com. XML-Schema Tutorial.
http://www.w3schools.com/Schema/default.asp

[17]Alejandro Pérez García. Manejo de certificados con keytool para la activación de SSL.
http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=securitySSLKeytool.

[18] Openssl.org. OpenSSL Manual.
http://www.openssl.org/docs/apps/openssl.html

[19] Chuidiang.com. Lectura y escritura de ficheros en Java.
http://chuwiki.chuidiang.org/index.php?title=Lectura_y_Escritura_de_Ficheros_en_Java

[20] XML Signature Syntax and Processing
http://www.w3.org/TR/xmldsig-core/

[21] Network Time Protocol Version 4: Protocol and Algorithms Specification.
http://tools.ietf.org/html/rfc5905

[22] Apache Software Foundation. Apache Commons Net Library documentation.
http://commons.apache.org/net/

[23] Ministerio de Defensa de España. Servicio “hora” del Real Observatorio de la Armada.
http://www.armada.mde.es/ArmadaPortal/page/Portal/ArmadaEspannola/ciencia_observatorio/06_Hora

[24] Ministerio de Economía y Hacienda. Política de Firma versión 3.1 Formato Facturae.
http://www.facturae.es/politica_de_firma_formato_facturae/politica_de_firma_formato_facturae_v3_1.pdf

[25] Jdom.org. JDOM 2.0.2 API Specification
http://www.jdom.org/docs/apidocs/

[26] Alex Guerra. Java y XML: JDOM. Agosto 2008
http://www.latascadexela.es/2008/08/java-y-xml-jdom.html

[27]Protocolo de transferencia de hipertexto (HTTP).
http://www.w3.org/Protocols/rfc2616/rfc2616.html

APÉNDICES

APÉNDICE A: MANUAL DE USUARIO

Arranque del sistema

Para el arranque del sistema se necesita abrir tres ventanas de consola, cada una correspondiente a uno de los directorios de instalación (ver Apéndice B: Instalación). Por ello abriremos las siguientes ventanas de consola (independientemente de en qué equipo esté cada directorio instalado):

- Ventana de consola en el directorio CERTILOC
- Ventana de consola en el directorio TSA
- Ventana de consola en el directorio Subject

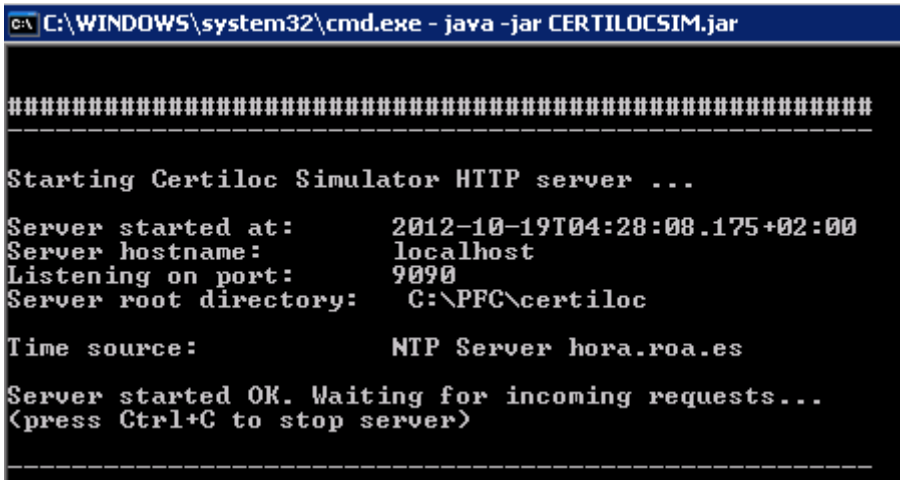
Para el caso de haber instalado en sistema Windows, las dimensiones recomendadas de cada ventana son las siguientes:

- Fuente de mapa de bits, tamaño 8x12
- Tamaño del búfer de pantalla: 137 ancho, 300 alto.
- Tamaño de la ventana: 137 ancho, 59 alto.

Arranque del servidor CERTILOC

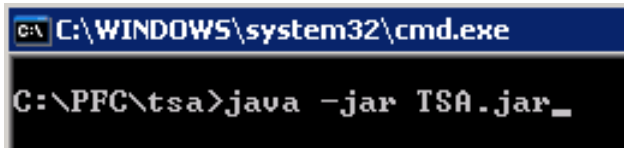
1) Teclear el comando siguiente: java –jar CERTILOCSIM.jar , o el nombre que se haya puesto al jar de arranque de certiloc (Ver Apéndice C Mantenimiento):

2) Comprobar que se borrará el contenido de la consola y en su lugar aparecerá un mensaje que termina con el texto “Served started OK”:

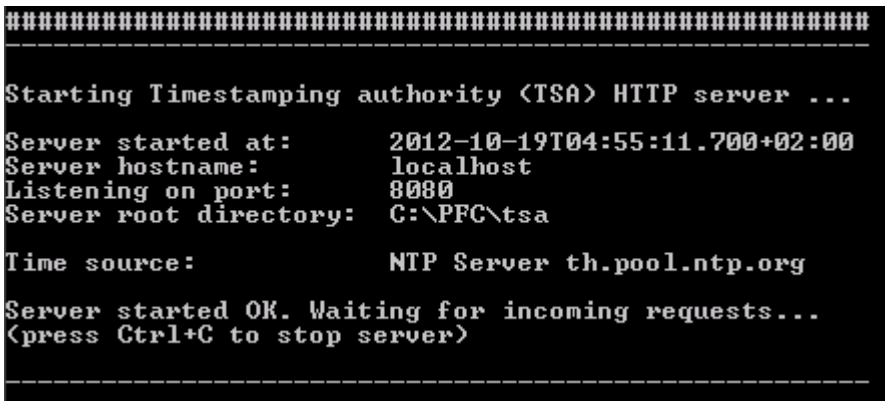


Arranque del servidor TSA

1) Teclear el comando siguiente: java –jar TSA.jar , o el nombre que se haya puesto al jar de arranque de la TSA (Ver Apéndice C Mantenimiento):

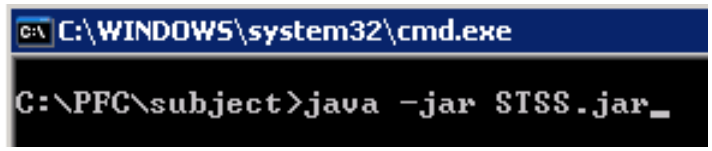


2) Comprobar que se borrará el contenido de la consola y en su lugar aparecerá un mensaje que termina con el texto “Served started OK”:

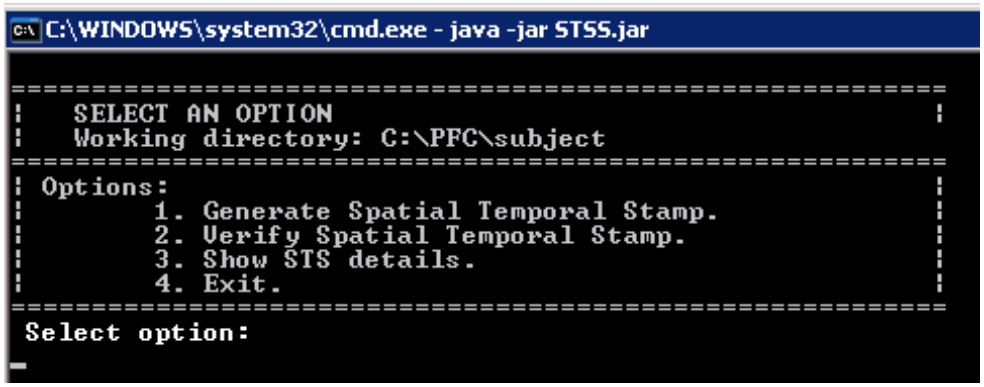


Arranque del software del Sistema de Sellador-Validador

1) Teclear el comando siguiente: java –jar STSS.jar, o el nombre que se haya puesto al jar de arranque del STSGV (Ver Apéndice C Mantenimiento):



2) Comprobar que se borrará el contenido de la consola y en su lugar aparecerá el menú principal del programa, bien se trate de la opción de Administración o la de usuario final (como en la siguiente figura):



Manual de usuario para usuario final

Visión general

Este manual de usuario va dirigido a los usuarios finales de este software, y como se verá este menú se centrará únicamente en ofrecer al usuario la mínima interacción e información de cómo se genera el Sello Espacio Temporal. El usuario solo tiene las opciones de generar el Sello Espacio Temporal, verificarlo y visualizarlo, pidiéndosele únicamente que proporcione el CET requerido y el Mensaje para el que generar el Sello Espacio Temporal.

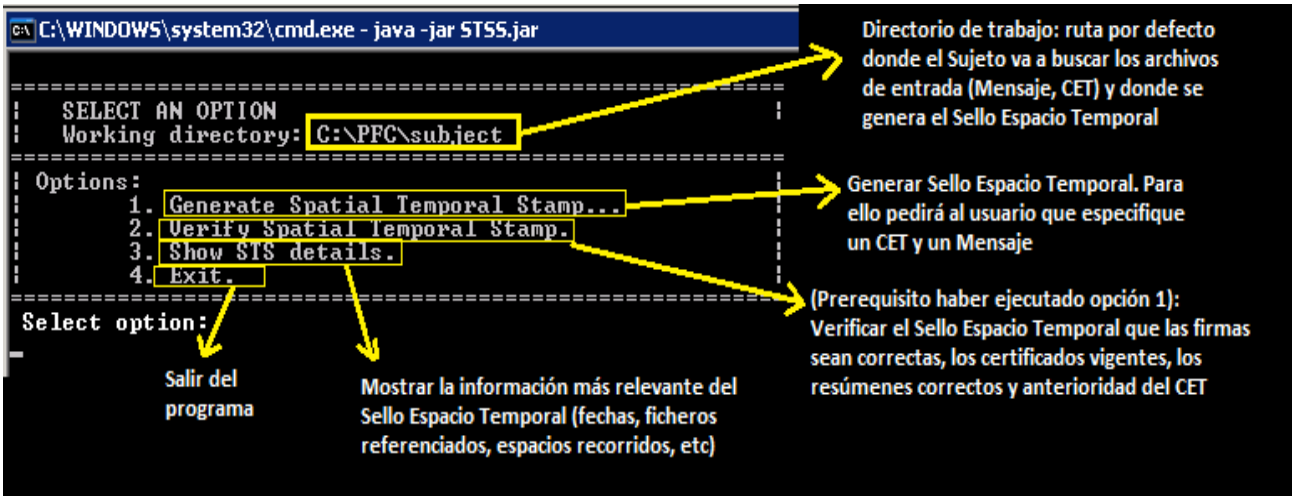


Figura: Menu principal de usuario final

Working directory: Hace referencia al directorio donde se está ejecutando el STSGV, es el directorio donde por defecto se encontrarán los archivos necesarios y donde se generará finalmente el Sello Espacio temporal.

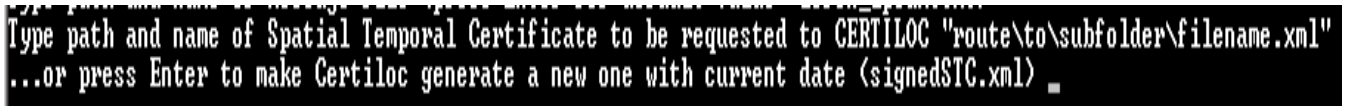
Opción 1: Generate Spatial Temporal Stamp (Generar Sello Espacio Temporal)

Esta opción de menu (1) debe ejecutarse antes que cualquiera de las otras dos opciones para poder generar un Sello Espacio Temporal.

El programa pedirá que tecleemos la ruta y nombre del archivo de Mensaje. Se admite todo tipo de archivos sean binarios o de texto, El programa está preparado para abortar la operación cuando este archivo exceda el límite de tamaño (ver sección Pruebas de tamaño de mensaje). Los directorios se indican con el carácter barra invertida (\).

En el ejemplo pulsamos enter para que el archivo sea el que se da por defecto (loren_ipsum.txt), el programa pedirá introducir la clave del almacén de claves y de la clave privada del Sujeto:

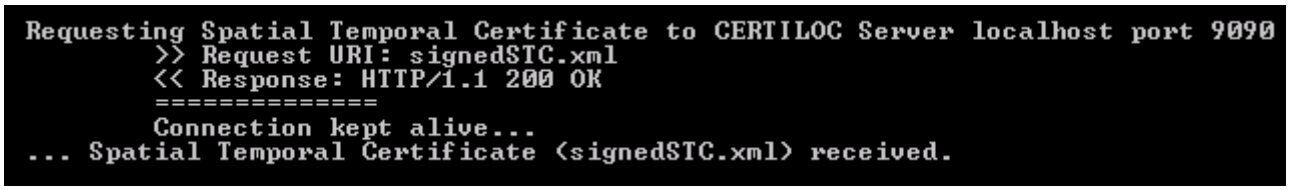
El programa pedirá que especifiquemos un Certificado Espacio Temporal a utilizar (o pulsamos enter para que nos de el archivo por defecto):



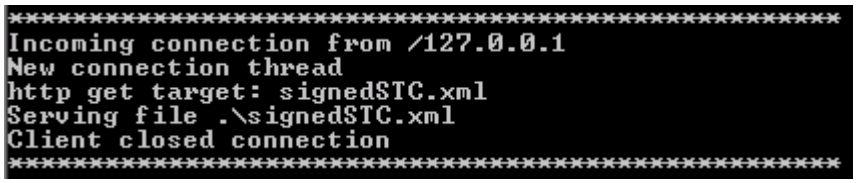
Para ello deberemos conocer de antemano la ruta y nombre del archivo CET a recuperar. Teclearemos la ruta relativa del archivo xml que contiene el Certificado Espacio Temporal. Esta ruta relativa parte del directorio de trabajo donde esté funcionando el servidor de certiloc. Por ejemplo para el sitio de instalación de ejemplo hay que teclear la ruta stcs\signedSTC.xml. Los directorios se indican con el carácter barra invertida (\).

Si pulsamos enter o tecleamos el nombre signedSTC.xml, CERTILOC generará un CET en el momento con la fecha actual y una localización espacial aleatoria.

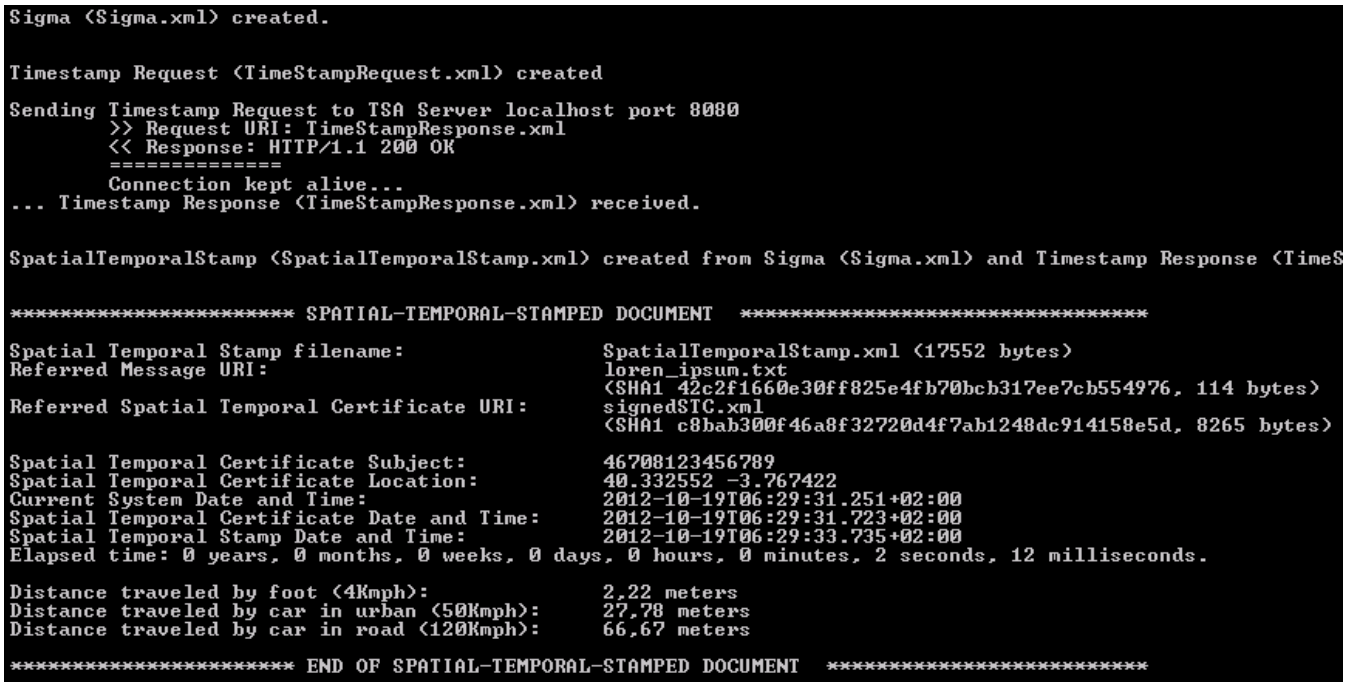
La salida de consola será la siguiente, indicando que el archivo se ha recuperado correctamente:



En la consola del servidor CERTILOC, aparecerán una serie de líneas indicando el tratamiento de la petición:



Por último el programa se conecta al servidor de la TSA para obtener la información de tiempo y a continuación crea el archivo físico del sello espacio temporal, para por último mostrarnos lo más relevante de su contenido.



En la consola de la TSA se mostrarán una serie de líneas informando de el tratamiento de la petición:

```
**********************************************************
Incoming connection from /127.0.0.1
New connection thread
Incoming entity content 1700 bytes.
Timestamp Request (TimeStampRequest.xml) file created
Timestamp Response (TimeStampResponse.xml) file created with serial number 7...
Serving file .\TimeStampResponse.xml
Client closed connection
**********************************************************
```

Opción 2: Verify Spatial Temporal Stamp (Verificar Sello Espacio Temporal)

En primer lugar se mostrará por consola los datos principales del Sello Espacio Temporal:

```
********************** SPATIAL-TEMPORAL-STAMPED DOCUMENT  ********************************

Spatial Temporal Stamp filename:                SpatialTemporalStamp.xml (17552 bytes)
Referred Message URI:                           loren_ipsum.txt
                                                (SHA1 42c2f1660e30ff825e4fb70bcb317ee7cb554976, 114 bytes)
Referred Spatial Temporal Certificate URI:      signedSTC.xml
                                                (SHA1 c8bab300f46a8f32720d4f7ab1248dc914158e5d, 8265 bytes)

Spatial Temporal Certificate Subject:           46708123456789
Spatial Temporal Certificate Location:          40.332552 -3.767422
Current System Date and Time:                   2012-10-19T06:29:31.251+02:00
Spatial Temporal Certificate Date and Time:     2012-10-19T06:29:31.723+02:00
Spatial Temporal Stamp Date and Time:           2012-10-19T06:29:33.735+02:00
Elapsed time: 0 years, 0 months, 0 weeks, 0 days, 0 hours, 0 minutes, 2 seconds, 12 milliseconds.

Distance traveled by foot (4Kmph):              2,22 meters
Distance traveled by car in urban (50Kmph):     27,78 meters
Distance traveled by car in road (120Kmph):     66,67 meters

********************** END OF SPATIAL-TEMPORAL-STAMPED DOCUMENT  **************************
```

Verificaciones respecto al Sello Espacio Temporal (Spatial Temporal Stamp Validations):

El programa verificará las firmas tanto de Sigma como de la TSA dentro del Sello Espacio Temporal:

```
 ------------------ Spatial Temporal Stamp Validations: -------------------------------

 OK Valid signature element 'SigmaSignatureElement'
          from file SpatialTemporalStamp.xml

 OK Valid signature element 'TimeStampResponseSignatureElement'
          from file SpatialTemporalStamp.xml
```

Se verificará que las rutas de certificación de las firmas son válidas y que los certificados que las componen no están caducadas:

```
OK Certification path /tsp:SpatialTemporalStamp/ds:Signature
          from file SpatialTemporalStamp.xml

OK Certification path /tsp:SpatialTemporalStamp/tsp:TimeStampToken/ds:Signature
          from file SpatialTemporalStamp.xml
```

Se verificará que el Sello Espacio Temporal sea posterior al instante marcado en el Certificado Espacio Temporal:

```
Spatial Temporal Certificate Date:       2012-10-19T06:29:31.723+02:00
Spatial Temporal Stamp Date:             2012-10-19T06:29:33.735+02:00
OK Spatial Temporal Stamp was Generated after Spatial Temporal Certificate
```

Se verificará que los valores resumen calculados en el Sello Espacio Temporal sean correctos:

```
OK hash of Message File (loren_ipsum.txt) in Spatial Temporal Stamp (SpatialTemporalS
OK hash of Spatial Temporal Certificate File (signedSTC.xml) in Spatial Temporal Stam
OK hash of Reference Element (URI=#SigmaSignatureElement) in Spatial Temporal Stamp (
OK hash of Message Imprints Element in Spatial Temporal Stamp (SpatialTemporalStamp.x
```

Verificaciones respecto al Certificado Espacio Temporal (Spatial Temporal Certificate validations):

Se verificará la firma del Certificado Espacio Temporal, así como la ruta de certificación del mismo:

```
------------------ Spatial Temporal Certificate Validations: ---------------------

OK Valid signature element 'STCSignature'
         from file signedSTC.xml

OK Certification path /saml:Assertion/ds:Signature
         from file signedSTC.xml
```

Verificaciones respecto a la respuesta de la TSA (Timestamp Response Validations)

Se verificará la firma de la respuesta de la TSA (TimeStamp Response), así como la ruta de certificación del mismo:

```
------------------ Timestamp Response Validations ------------------------------

OK Valid signature element 'TimeStampResponseSignatureElement'
         from file SpatialTemporalStamp.xml

OK Certification path /tsp:SpatialTemporalStamp/tsp:TimeStampToken/ds:Signature
         from file SpatialTemporalStamp.xml
```

Opción 3: Show Spatial Temporal Stamp details (mostrar detalles del Sello Espacio Temporal)

Para ejecutar esta opción antes se deberá haber ejecutado la opción del menú 1 (generar Sello espacio Temporal). Se mostrarán los detalles más relevantes del Sello Espacio Temporal:

```
********** SPATIAL-TEMPORAL-STAMPED DOCUMENT  **********

Spatial Temporal Stamp filename:                SpatialTemporalStamp.xml (17552 bytes)
Referred Message URI:                           loren_ipsum.txt
                                                (SHA1 42c2f1660e30ff825e4fb70bcb317ee7cb554976, 114 bytes)
Referred Spatial Temporal Certificate URI:      signedSTC.xml
                                                (SHA1 c8bab300f46a8f32720d4f7ab1248dc914158e5d, 8265 bytes)

Spatial Temporal Certificate Subject:           46708123456789
Spatial Temporal Certificate Location:          40.332552 -3.767422
Current System Date and Time:                   2012-10-19T06:34:24.411+02:00
Spatial Temporal Certificate Date and Time:     2012-10-19T06:29:31.723+02:00
Spatial Temporal Stamp Date and Time:           2012-10-19T06:29:33.735+02:00
Elapsed time: 0 years, 0 months, 0 weeks, 0 days, 0 hours, 0 minutes, 2 seconds, 12 milliseconds.

Distance traveled by foot (4Kmph):              2,22 meters
Distance traveled by car in urban (50Kmph):     27,78 meters
Distance traveled by car in road (120Kmph):     66,67 meters

********** END OF SPATIAL-TEMPORAL-STAMPED DOCUMENT  **********
```

Nombre del fichero del Sello Espacio Temporal. El nombre del fichero a generar estará determinado por el parámetro de configuración FILES.SPATIALTEMPORALSTAMP.FILENAME:

```
Spatial Temporal Stamp filename:                SpatialTemporalStamp.xml (17552 bytes)
```

Ruta del fichero de Mensaje referenciado. El nombre del fichero de Mensaje para el que se creó el Sello Espacio Temporal, junto con el cálculo del resumen del mismo en hexadecimal y su tamaño en bytes:

```
Referred Message URI:                           loren_ipsum.txt
                                                (SHA1 42c2f1660e30ff825e4fb70bcb317ee7cb554976, 114 bytes)
```

Ruta del fichero de Certificado Espacio Temporal referenciado. El nombre del fichero del Certificado Espacio Temporal para el que se creó el Sello Espacio Temporal, junto con el cálculo del resumen del mismo en hexadecimal y su tamaño en bytes:

```
Referred Spatial Temporal Certificate URI:      signedSTC.xml
                                                (SHA1 c8bab300f46a8f32720d4f7ab1248dc914158e5d, 8265 bytes)
```

Información de localización y de identificación del Certificado Espacio Temporal:

```
Spatial Temporal Certificate Subject:           46708123456789
Spatial Temporal Certificate Location:          40.332552 -3.767422
```

Fecha y hora del equipo donde se está ejecutando el programa del Sellador-Validador. Esta fecha no aparece como tal en el Sello Espacio Temporal, solo se incluye a título informativo:

```
Current System Date and Time:                   2012-10-19T06:34:24.411+02:00
```

Fechas del Certificado Espacio Temporal y del Sello Espacio Temporal, y tiempo transcurrido entre estas:

```
Spatial Temporal Certificate Date and Time:     2012-10-19T06:29:31.723+02:00
Spatial Temporal Stamp Date and Time:           2012-10-19T06:29:33.735+02:00
Elapsed time: 0 years, 0 months, 0 weeks, 0 days, 0 hours, 0 minutes, 2 seconds, 12 milliseconds.
```

Distancia recorrida en el tiempo transcurrido entre las fechas del Certificado Espacio Temporal y el Sello Espacio Temporal:

```
Distance traveled by foot (4Kmph):              2,22 meters
Distance traveled by car in urban (50Kmph):     27,78 meters
Distance traveled by car in road (120Kmph):     66,67 meters
```

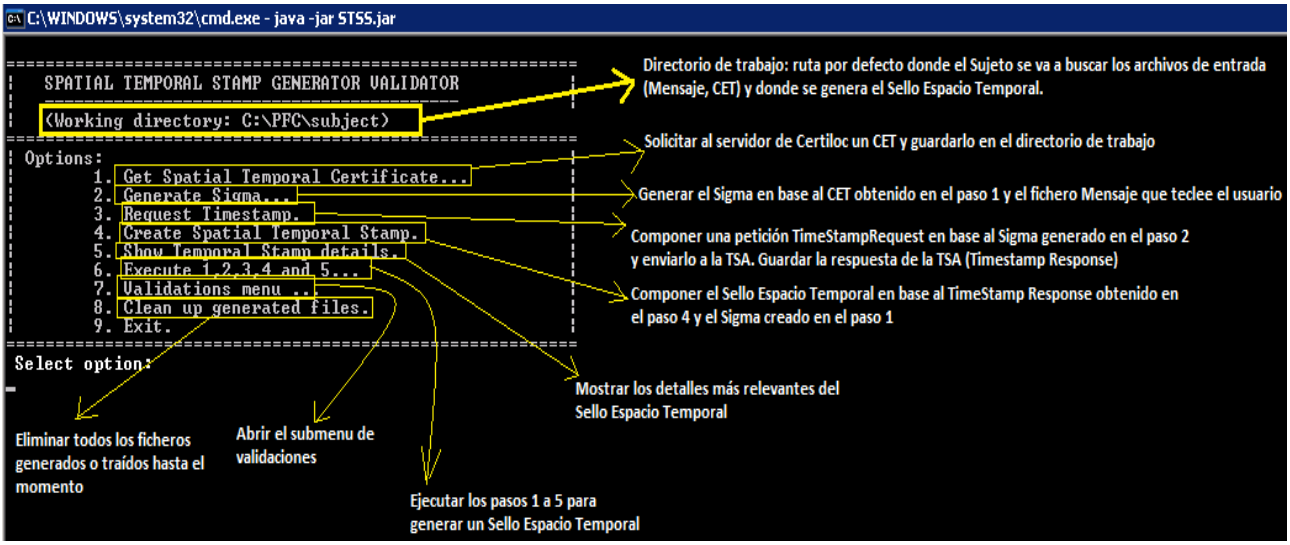
Manual de usuario para modo Administrador

Visión general

El menú de usuario para modo Administrador es más completo porque permite ejecutar las operaciones necesarias para componer un Sello Espacio Temporal separadamente cada una. Los ficheros intermedios creados en los sucesivos pasos permanecen:

- Opción 1: Se trae el CET (por defecto signedSTC.xml)
- Opción 2: Se crea Sigma.xml
- Opción 3: Se crean los TimeStampRequest.xml y TimeStampResponse.xml
- Opción 4: Se crea el SpatialTemporalStamp.xml

Hay que tener en cuenta que dichas operaciones deben ejecutarse en el orden adecuado porque cada una requiere los archivos generado en la operación anterior. Por otro lado, si ejecutamos por ejemplo el paso 1 despues de haber ejecutado la serie de pasos 1 a 4 en un momento anterior, los ficheros presentes en el sistema como consecuencia de la anterior ejecución de pasos estarán "desactualizados". Dicho de otra forma, si ejecutamos los pasos 1 a 4 y después ejecutamos de nuevo el paso 1, los ficheros que quedaron de los pasos 2, 3 y 4 estarán desactualizados respecto al nuevo paso 1 ejecutado.



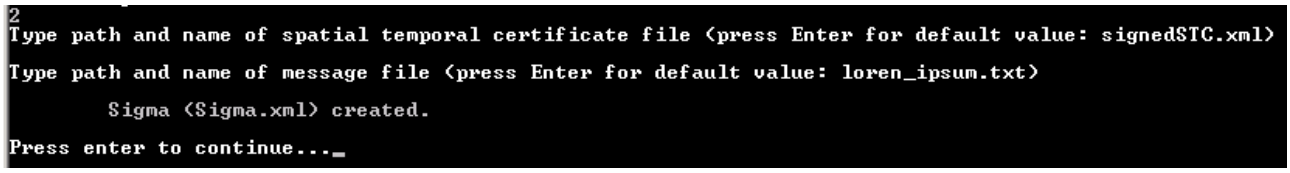
Opción 1: Get Spatial Temporal Certificate (Obtener Certificado Espacio Temporal)

Esta opción nos permite solicitar a CERTILOC un determinado Certificado Espacio Temporal. Si pulsamos enter nos devolverá el CET por defecto (signedSTC.xml). Si tecleamos un nombre de CET y este nos encuentra en el servidor, nos mostrará un mensaje indicándonos que tecleemos uno correcto hasta que lo hagamos. A continuación nos mostrará unas líneas indicándonos que la petición ha sido realizada con éxito:

```
Type path and name of Spatial Temporal Certificate to be requested to CERTILOC "route\to\subfolder\filename.xml"
...or press Enter to make Certiloc generate a new one with current date (signedSTC.xml)

        Requesting Spatial Temporal Certificate to CERTILOC Server localhost port 9090
                >> Request URI: signedSTC.xml
                << Response: HTTP/1.1 200 OK
                ==============
                Connection kept alive...
        ... Spatial Temporal Certificate (signedSTC.xml) received.

Press enter to continue...
```

Opción 2: Generate Sigma (Generar Sigma)

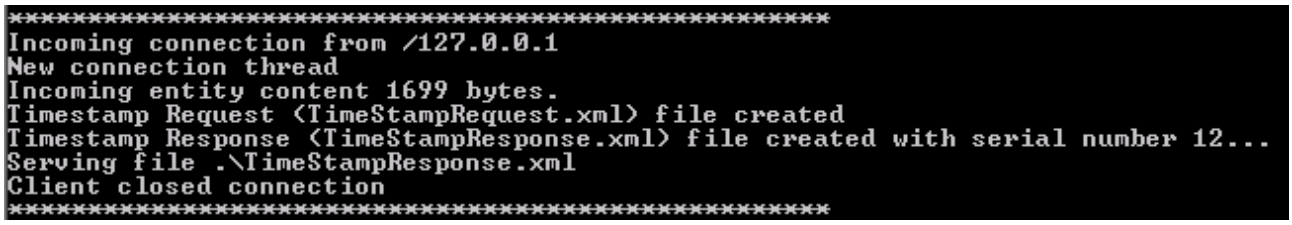
Esta opción nos pedirá que tecleemos el nombre de un CET (típicamente teclearemos el nombre del CET obtenido en el paso anterior), y nos pedirá que tecleemos un fichero de Mensaje. Si el CET o Mensaje tecleado no existe, nos lo pedirá hasta que tecleemos uno válido. Con estos dos ficheros compondrá el fichero Sigma.xml. Los directorios se indican con el carácter barra invertida (\):



Opción 3: Request TimeStamp (Pedir Sello de Tiempo)

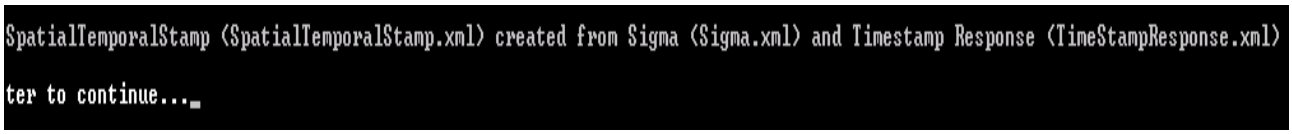
Esta opción tomará el fichero Sigma.xml generado en el paso anterior y compondrá un fichero TimeStampRequest.xml que enviará al servidor de la TSA en una petición Http. El servidor TSA procesará dicha petición y devolverá un fichero TimeStampResponse.xml como respuesta:

En la consola de la TSA se verá el procesamiento de dicha petición:



Opción 4: Create Spatial Temporal Stamp (Crear Sello Espacio Temporal)

Esta opción hace que se componga el Sello Espacio Temporal en base a los ficheros Sigma.xml y TimeStampResponse.xml obtenidos en los pasos 2 y 3 respectivamente:



Opción 5: Show Spatial Temporal Stamp details (mostrar detalles del Sello Espacio Temporal)

Para ejecutar esta opción antes se deberá haber ejecutado o bien las opciones de menú de 1 a 4 en ese orden, o bien la opción de menu 6 (Generar Sello Espacio Temporal). Se mostrarán los detalles más relevantes del Sello Espacio Temporal:

```
********************** SPATIAL-TEMPORAL-STAMPED DOCUMENT  ********************************

Spatial Temporal Stamp filename:                SpatialTemporalStamp.xml (17552 bytes)
Referred Message URI:                           loren_ipsum.txt
                                                (SHA1 42c2f1660e30ff825e4fb70bcb317ee7cb554976, 114 bytes)
Referred Spatial Temporal Certificate URI:      signedSTC.xml
                                                (SHA1 c8bab300f46a8f32720d4f7ab1248dc914158e5d, 8265 bytes)

Spatial Temporal Certificate Subject:           46708123456789
Spatial Temporal Certificate Location:          40.333255 -3.767046
Current System Date and Time:                   2012-10-19T06:40:23.703+02:00
Spatial Temporal Certificate Date and Time:     2012-10-19T06:38:43.283+02:00
Spatial Temporal Stamp Date and Time:           2012-10-19T06:29:33.735+02:00
Elapsed time: 0 years, 0 months, 0 weeks, 0 days, 0 hours, -9 minutes, -9 seconds, -548 milliseconds.

Distance traveled by foot (4Kmph):              -610 meters
Distance traveled by car in urban (50Kmph):     -7.625 meters
Distance traveled by car in road (120Kmph):     -18.300 meters

********************** END OF SPATIAL-TEMPORAL-STAMPED DOCUMENT  **************************
```

Nombre del fichero del Sello Espacio Temporal. El nombre del fichero a generar estará determinado por el parámetro de configuración FILES.SPATIALTEMPORALSTAMP.FILENAME:

```
Spatial Temporal Stamp filename:                SpatialTemporalStamp.xml (17552 bytes)
```

Ruta del fichero de Mensaje referenciado. El nombre del fichero de Mensaje para el que se creó el Sello Espacio Temporal, junto con el cálculo del resumen del mismo en hexadecimal y su tamaño en bytes:

```
Referred Message URI:                           loren_ipsum.txt
                                                (SHA1 42c2f1660e30ff825e4fb70bcb317ee7cb554976, 114 bytes)
```

Ruta del fichero de Certificado Espacio Temporal referenciado. El nombre del fichero del Certificado Espacio Temporal para el que se creó el Sello Espacio Temporal, junto con el cálculo del resumen del mismo en hexadecimal y su tamaño en bytes:

```
Referred Spatial Temporal Certificate URI:      signedSTC.xml
                                                (SHA1 c8bab300f46a8f32720d4f7ab1248dc914158e5d, 8265 bytes)
```

Información de localización y de identificación del Certificado Espacio Temporal:

```
Spatial Temporal Certificate Subject:           46708123456789
Spatial Temporal Certificate Location:          40.332552 -3.767422
```

Fecha y hora del equipo donde se está ejecutando el programa del Sellador-Validador. Esta fecha no aparece como tal en el Sello Espacio Temporal, solo se incluye a título informativo:

```
Current System Date and Time:                   2012-10-19T06:34:24.411+02:00
```

Fechas del Certificado Espacio Temporal y del Sello Espacio Temporal, y tiempo transcurrido entre estas:

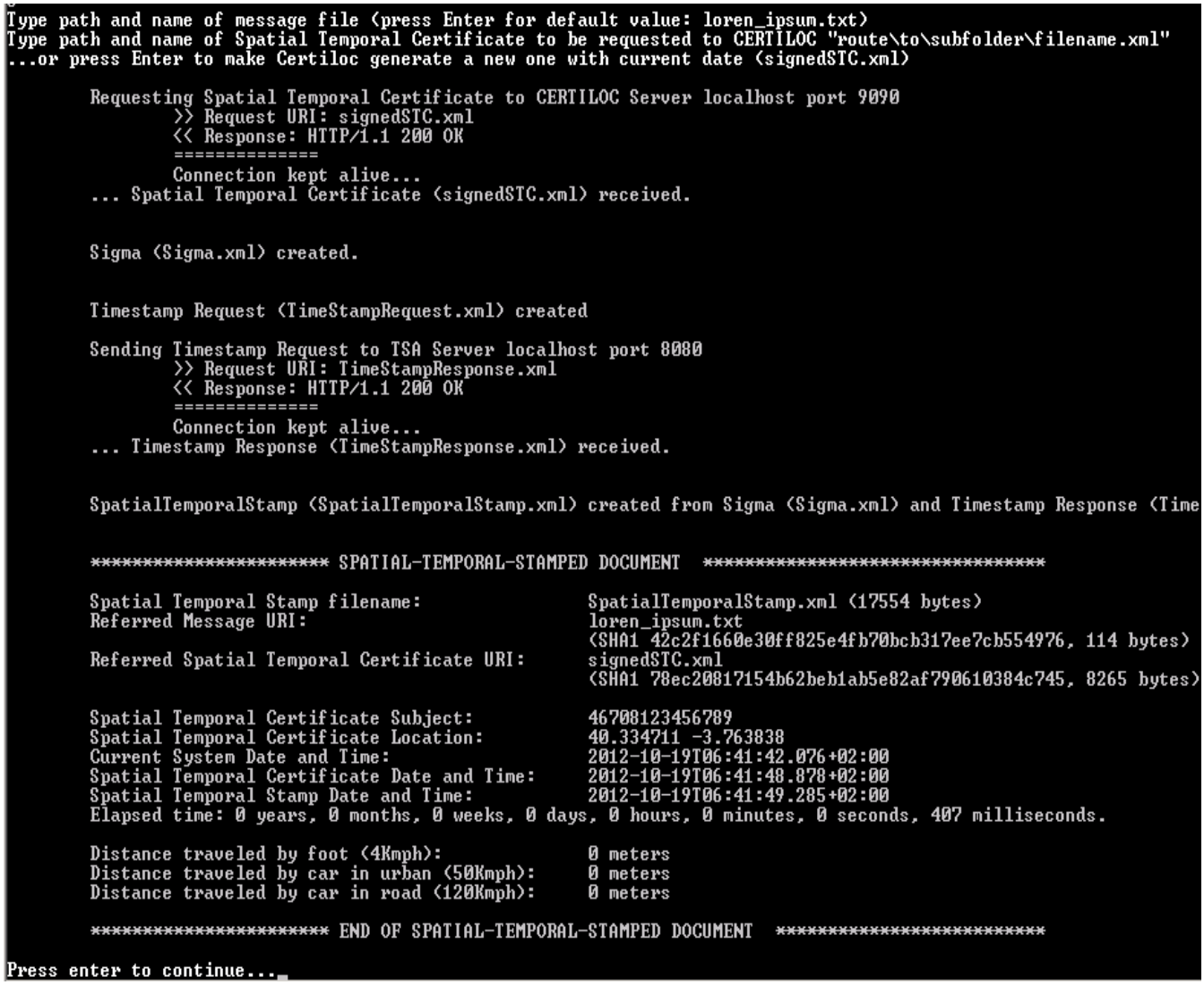
```
Spatial Temporal Certificate Date and Time:     2012-10-19T06:29:31.723+02:00
Spatial Temporal Stamp Date and Time:           2012-10-19T06:29:33.735+02:00
Elapsed time: 0 years, 0 months, 0 weeks, 0 days, 0 hours, 0 minutes, 2 seconds, 12 milliseconds.
```

Distancia recorrida en el tiempo transcurrido entre las fechas del Certificado Espacio Temporal y el Sello Espacio Temporal:

```
Distance traveled by foot (4Kmph):              2,22 meters
Distance traveled by car in urban (50Kmph):     27,78 meters
Distance traveled by car in road (120Kmph):     66,67 meters
```

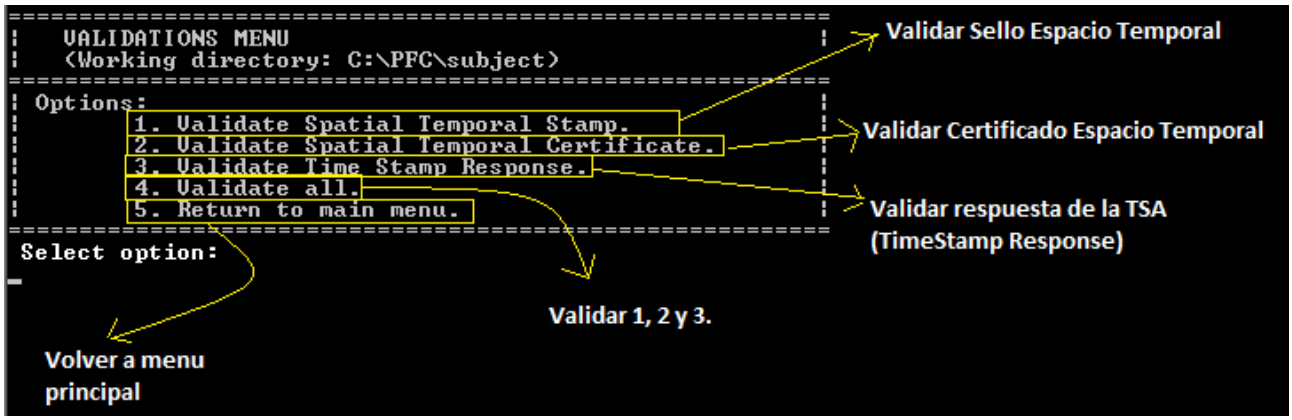
Opción 6: Execute 1,2,3,4 and 5 (Ejecutar opciones 1 a 5)

Esta opción ejecuta secuencialmente los pasos 1 a 5:



Opción 7 Validations menu (Menu de validaciones)

Esta opción abre un nuevo submenú de Validaciones que nos permite ejecutar diversas validaciones. Para ello se han tenido que ejecutar los pasos 1 a 4 o de lo contrario el programa mostrará un mensaje de error informando de dicho requisito.



Opción 7.1 Validate Spatial Temporal Stamp (Validar Sello Espacio Temporal)

Esta opción seguirá el proceso descrito en las secciones 7.2 y 7.3 para validar el contenido del Sello Espacio Temporal en el que están implicadas varias firmas:

La firma por parte del Sujeto de Sigma:

```
OK Valid signature element 'SigmaSignatureElement'
        from file SpatialTemporalStamp.xml
```

La firma por parte de la TSA de la respuesta (TimeStamp Response)

```
OK Valid signature element 'TimeStampResponseSignatureElement'
        from file SpatialTemporalStamp.xml
```

La ruta de certificación de las credenciales del Sujeto en la firma de Sigma:

```
OK Certification path /tsp:SpatialTemporalStamp/ds:Signature
        from file SpatialTemporalStamp.xml
```

La ruta de certificación de las credenciales de la TSA en la firma de la respuesta:

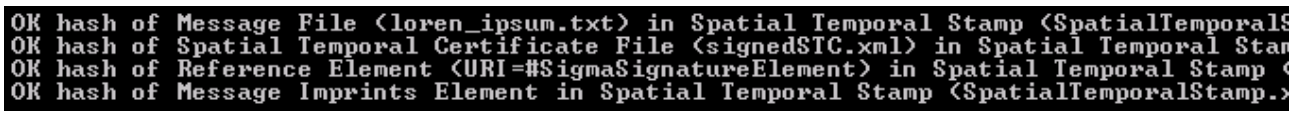
```
OK Certification path /tsp:SpatialTemporalStamp/tsp:TimeStampToken/ds:Signature
        from file SpatialTemporalStamp.xml
```

Se verificará que el Sello Espacio Temporal sea posterior al instante marcado en el Certificado Espacio Temporal:

```
Spatial Temporal Certificate Date:       2012-10-19T06:29:31.723+02:00
Spatial Temporal Stamp Date:             2012-10-19T06:29:33.735+02:00
OK Spatial Temporal Stamp was Generated after Spatial Temporal Certificate
```

Que los resúmenes incluidos en el Sello Espacio Temporal sean correctos respecto a la información a la que se refieren:

- El resumen del fichero Mensaje
- El resumen del fichero Certificado Espacio Temporal
- El resumen del elemento ds:Reference del Sello Espacio Temporal
- El resumen del elemento Message Imprints



APENDICE B: MANUAL DE INSTALACIÓN

Requisitos hardware y software

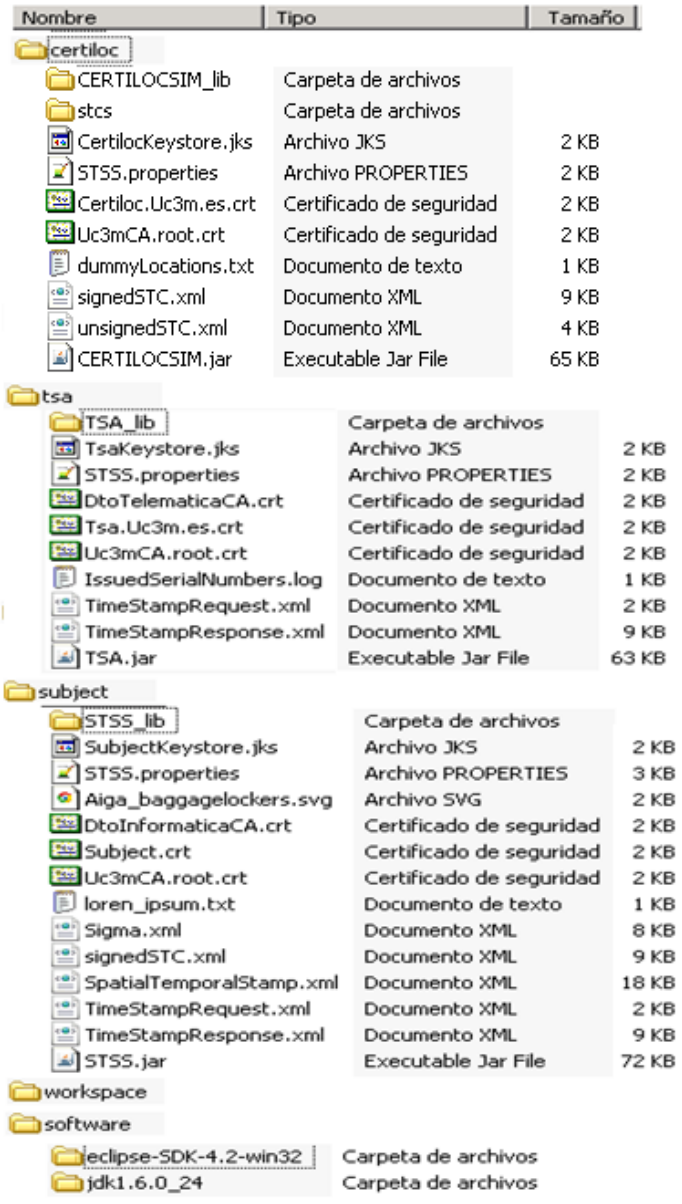
El requisito mínimo de funcionamiento es un ordenador con al menos 1.6Ghz de procesador y 256Mb de RAM. Se puede instalar en un sistema operativo Windows o Linux, ya que se ejecuta en el entorno de una máquina virtual java.

Para más información ver la sección Configuración de mantenimiento Recomendada.

Se requiere tener permisos de escritura para poder copiar la estructura de archivos al equipo de instalación y tener permisos para modificar las variables de entorno del sistema.
Aunque este software está desarrollado en Java, no se requiere presencia de JVM (java virtual machine) en el equipo de instalación puesto que con este software ya se entrega Java SDK 1.6.0u24.

Descripción del contenido del empaquetado de instalación

El paquete de instalación contiene lo siguiente:



DIRECTORIO/ARCHIVO		EXPLICACIÓN
CERTILOC		
	Stcs	Subcarpeta de ejemplo (se puede borrar)
	CERTILOCSIM_lib	Librerías de las que depende el CERTILOCSIM.jar Se generará automáticamente al exportar el jar de CERTILOCSIM.jar con la opción "Copy required libraries into a sub-folder next to the generated Jar"
	Uc3mCA.root.crt	Certificados de clave pública de CERTILOC
	CERTILOC.Uc3m.es.crt	
	CERTILOCKeystore.jks	Almacén de claves de certiloc
	STSS.properties	Archivo de configuración de certiloc
	dummyLocations.txt	Archivo de localizaciones espaciales para la generación de Certificados Espacio Temporales.
	unsignedSTC.xml	Certificado Espacio Temporal sin firmar de ejemplo (se puede borrar)
	signedSTC.xml	Certificado Espacio Temporal firmado de ejemplo (se puede borrar)
	CERTILOCSIM.jar	Empaquetado ejecutable de certiloc
TSA		
	TSA_lib	Librerías de las que depende el TSA.jar Se generará automáticamente al exportar el jar de TSA.jar con la opción "Copy required libraries into a sub-folder next to the generated Jar"
	TsaKeyStore.jks	Almacén de claves de la TSA
	STSS.properties	Archivo de configuración de la TSA
	DtoTelematicaCA.crt	Certificados de clave pública de la TSA
	TSA.Uc3m.es.crt	
	Uc3mCA.root.crt	
	IssuedSerialNumbers.log	Log de los serial numbers de las respuestas ya emitidas por la TSA. Inicialmente su único contenido válido es un carácter 0 seguido de un carácter retorno de carro.
	TimeStampRequest.xml	TimeStampRequest de ejemplo, será el TimeStampRequest de la última petición recibida (se puede borrar)
	TimeStampResponse.xml	TimeStampResponse de ejemplo, será el TimeStampResponse último devuelto por la TSA (se puede borrar)
	TSA.jar	Empaquetado ejecutable de la TSA
SUBJECT		
	STSS_lib	Librerías de las que depende el STSS.jar Se generará automáticamente al exportar el jar de STSS.jar con la opción "Copy required libraries into a sub-folder next to the generated Jar"
	SubjectKeystore.jks	Almacén de claves del Sellador Validador.
	STSS.properties	Archivo de configuración del Sellador Validador.
	Aiga_baggagelockers.svg	Archivo de mensaje de ejemplo dejado aquí para demostrar que el Sellador Validador funciona con archivos binarios de imágenes (se puede borrar)
	DtoInformaticaCA.crt	Certificados de clave pública del Sellador Validador
	Subject.crt	
	Uc3mCA.root.crt	
	loren_ipsum.txt	Archivo de mensaje de ejemplo de tipo texto (se puede borrar)

	Sigma.xml	Archivo de Sigma de ejemplo, ha quedado aquí porque la última vez que se ejecutó el programa fue en modo administrador. Se puede borrar
	signedSTC.xml	Certificado Espacio Temporal firmado de ejemplo (se puede borrar)
	SpatialTemporalStamp.xml	Sello Espacio Temporal de ejemplo. Se puede borrar.
	TimeStampRequest.xml	TimeStampRequest de ejemplo, se puede borrar
	TimeStampResponse.xml	TimeStampResponse de ejemplo, se puede borrar
	STSS.jar	Empaquetado ejecutable del Sellador Validador
WORKSPACE		Código de la aplicación. Ver Sección 8.2 Arquitectura del Sistema para más información
SOFTWARE		
	eclipse-SDK-4.2-win32	Editor de código integrado
	jdk1.6.0_24	Máquina virtual Java (innecesario si ya se tiene máquina virtual instalada en los equipos donde se va a instalar el software)

Procedimiento de instalación

Instrucciones generales

Como requisito de funcionamiento, será necesario que el equipo de instalación tenga instalada una máquina virtual java versión 1.6 o superior. Para comprobarlo, abrir una ventana de consola y teclear el comando java-version, deberá devolver una serie de mensajes indicando la versión de java instalada si la hay. Típicamente en Windows la máquina virtual java estará instalada en c:\archivos de programa\java\.

Si no disponemos de una máquina virtual java, por comodidad en la carpeta software del equipo de instalación se provee del java development kit versión 1.6.0u24. Para que éste funcione deberán crearse si no existen una variable de sistema llamada JAVA_HOME que apunte a la ruta [directorio de instalación deseado]\software\ jdk1.6.0_24 y otra variable de sistema llamada path en el que incluyamos el valor %JAVA_HOME%\bin. Si no queremos utilizar ni la máquina virtual java que hay en Windows ni la que se proporciona en este paquete, podemos utilizar la que queramos siempre que establezcamos el valor de dichas dos variables de sistema

Por otro lado es importante reseñar que la hora obtenida desde un servidor NTP viene con respecto a la zona horaria UTC, siendo la entidad cliente de NTP la encargada de transformar dicha hora a la hora local del sistema de instalación, esto implica que tanto en el caso de CERTILOC como en el caso de la TSA, la hora del sistema deberá estar configurada de forma adecuada para evitar confusiones al interpretar las fechas que aparezcan, si bien todas las fechas que aparezcan tendrán aparejada la zona horaria de aplicación al mostrarse en pantalla y en los documentos implicados en el proceso.

Instalación del servidor simulador de CERTILOC

En el paquete de instalación del presente proyecto se tomará la carpeta llamada certiloc y se copiará a la ruta de directorio deseada dentro del equipo de instalación de certiloc. En el ejemplo propuesto, se ha instalado el directorio certiloc dentro de la ruta c:\PFC:

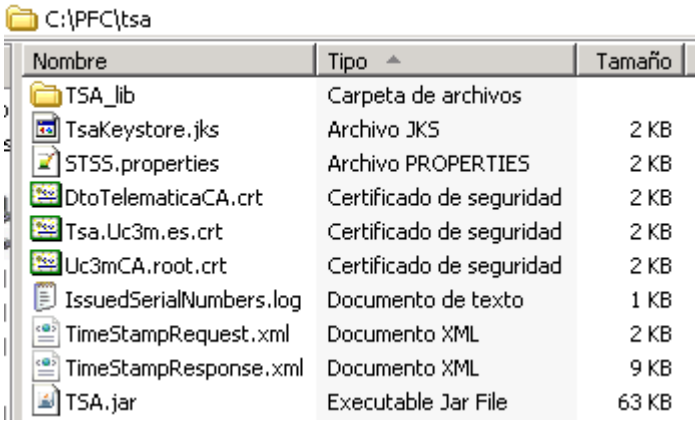
Una vez que tengamos copiada la carpeta certiloc en el equipo de instalación, tendremos que comprobar que los parámetros de configuración en el archivo certiloc\STSS.properties estén correctos. Especialmente críticos son los siguientes:

CERTILOCSIM.SERVER.PORT: El valor por defecto del puerto de escucha de certiloc es 9090. Comprobar que el equipo de instalación no tiene ningún firewall o cualquier software que bloquee dicho puerto o cambiar dicho valor a un puerto que esté abierto.

CERTILOCSIM.SERVER.ROOTDIRECTORYFORSERVINGFILES: Directorio raíz del servidor HTTP CERTILOC. Por defecto es el directorio donde se encuentra el archivo .jar ejecutable de CERTILOC (el directorio actual, es decir .\\, pero podemos poner una ruta relativa al mismo, teniendo cuidado de poner dos caracteres barra invertida. Por ejemplo ..\\otracarpeta\\certificados.

Instalación del servidor de la TSA

En el paquete de instalación del presente proyecto se tomará la carpeta llamada tsa y se copiará a la ruta de directorio deseada dentro del equipo de instalación de la TSA. En el ejemplo propuesto, se ha instalado el directorio tsa dentro de la ruta c:\PFC:



Una vez que tengamos copiada la carpeta tsa en el equipo de instalación, tendremos que comprobar que los parámetros de configuración en el archivo tsa\STSS.properties estén correctos. Especialmente críticos son los siguientes:

TSA.SERVER.PORT: El valor por defecto del puerto de escucha de la TSA es 8080. Comprobar que el equipo de instalación no tiene ningún firewall o cualquier software que bloquee dicho puerto o cambiar dicho valor a un puerto que esté abierto.

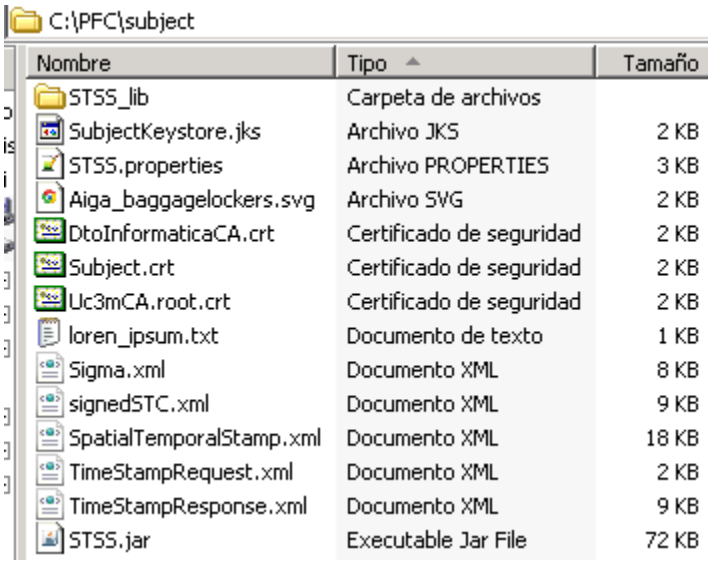
TSA.SERVER.ROOTDIRECTORYFORSERVINGFILES: Directorio raíz del servidor HTTP TSA. Por defecto es el directorio donde se encuentra el archivo .jar ejecutable de la TSA (el directorio actual, es decir .\\, pero podemos poner una ruta relativa al mismo, teniendo cuidado de poner dos caracteres barra invertida. Por ejemplo ..\\otracarpeta\\certificados.

TSA.TIME.ENABLENTPRETRIEVAL: En caso de que nuestro servidor no tenga acceso de red (por los motivos que sean como firewalls etc) a un servidor NTP, este valor deberá estar a false para que la fecha la obtenga de otras formas (ver la sección 4.5.4 Obtención de la fecha de la TSA)

Estado inicial del log de serial numbers emitidos: este archivo de log inicialmente deberá estar vacío para que el primer serial number a emitir sea 1.

Instalación del programa del Sellador Validador.

En el paquete de instalación del presente proyecto se tomará la carpeta llamada subject y se copiará a la ruta de directorio deseada dentro del equipo de instalación del Sellador Validador. En el ejemplo propuesto, se ha instalado el directorio subject dentro de la ruta c:\PFC:



Una vez que tengamos copiada la carpeta subject en el equipo de instalación, tendremos que comprobar que los parámetros de configuración en el archivo subject\STSS.properties estén correctos. Especialmente críticos son los siguientes:

CERTILOCSIM.SERVER.HOSTNAME: Dirección IP o nombre DNS del equipo que tenga instalado el servidor de CERTILOC. Deberá ser un equipo accesible a nivel de red.

CERTILOCSIM.SERVER.PORT: Puerto donde el servidor que tenga instalado CERTILOC está escuchando peticiones.

TSA.SERVER.HOSTNAME: Dirección IP o nombre DNS del equipo que tenga instalado el servidor de la TSA. Deberá ser un equipo accesible a nivel de red.

TSA.SERVER.PORT: Puerto donde el servidor que tenga instalado la TSA está escuchando peticiones.

SUBJECT.CONSOLEUI.MODE.ADVANCEDUI: Si el equipo donde vamos a instalar el software va a ser utilizado por usuarios normales, poner como valor false.

SUBJECT.CONSOLEUI.ASKFORCERTPASSWORD: Si el equipo donde vamos a instalar el software va a ser utilizado por usuarios normales, sería recomendable que el software pida la clave de acceso del almacén de claves del Sujeto, por lo que este parámetro debería estar a true

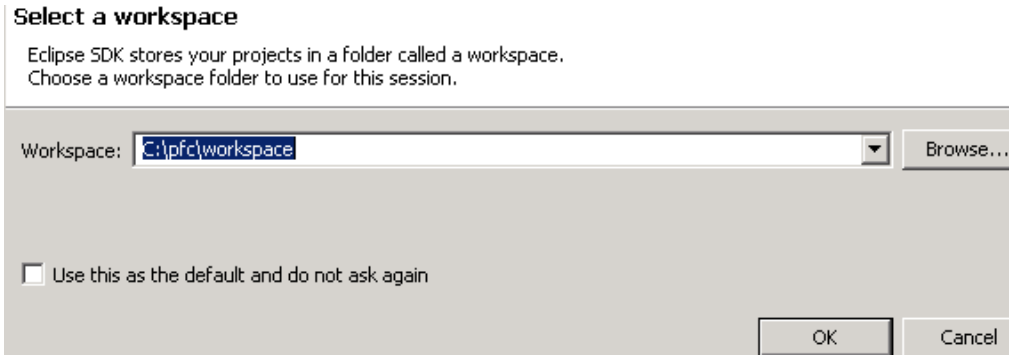
Instalación para mantenimiento del software

En el paquete de software estarán incluídos los directorios software y workspace.

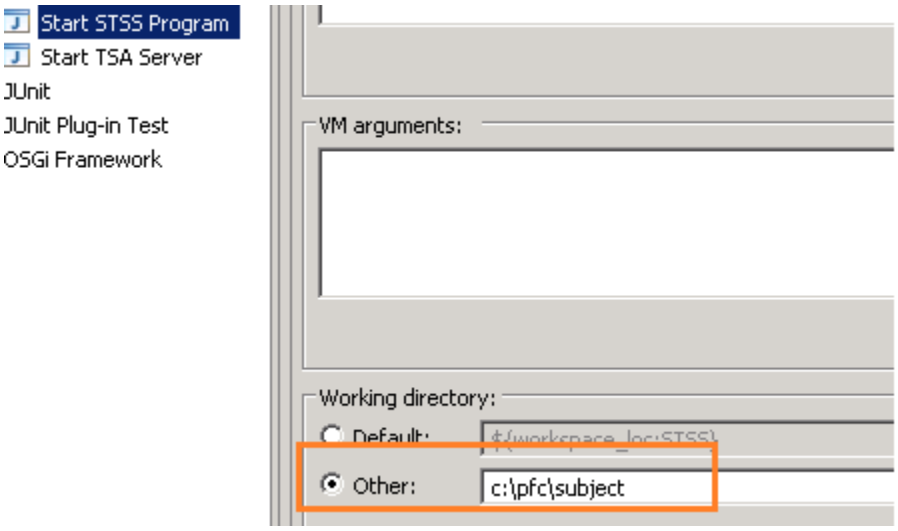
El directorio software contiene un editor eclipse y un jdk de java para el caso de que no tengamos alguno de los dos. Para más información consultar la sección "Apéndice C: Manual de mantenimiento- Configuración de mantenimiento recomendada".

El directorio workspace es el que contiene el código y está para ser visionado, editado y compilado por el editor Eclipse.

Para poder empezar a mantener el código de este software, hay que abrir el editor de eclipse y seleccionar este directorio workspace como entorno de trabajo:



Por último habrá que dejar preparadas las rutas por defecto de trabajo de cada uno de los proyectos contenidos en el workspace, para el caso de que queramos depurar código paso a paso.
Por ejemplo, para el proyecto STSS nos tenemos que ir a la run configuration correspondiente (Start STSS Program, que lo que hace es iniciar el menú principal del programa) y ponemos el directorio donde se va a ir a buscar los archivos manejados por la aplicación:



APÉNDICE C: MANUAL DE MANTENIMIENTO

Configuración de mantenimiento recomendada

Este software se ha desarrollado y probado en los siguientes equipos:

- Portátil Acer Aspire 3003LMi con Windows XP SP2. AMD Sempron 1.6Ghz. 32 bits. 1Gb RAM.
- Sobremesa con Windows XP SP2. AMD 1.6Ghz. 32 bits. 256Mb RAM.
- Portátil Dell latitude E5410 con Windows 7 Enterprise SP1. Intel Core i5 M520 2.4Ghz. 64 bits. 4Gb RAM
- Máquina virtual con Windows XP (Modo Windows XP del equipo Dell Latitude E5410). Windows XP Professional SP3. 2.4Ghz. 2Gb RAM.
- Portátil Acer Aspire 3003LMi con sistema operativo Linux Ubuntu versión.

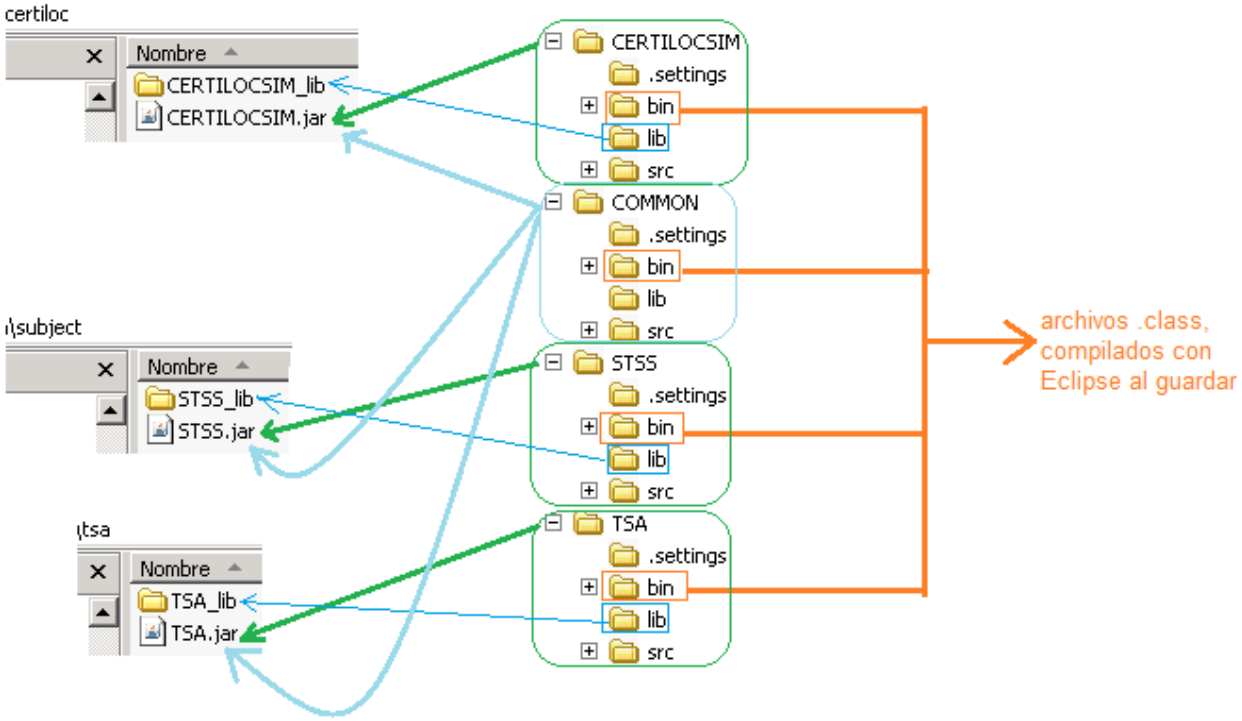
Por otro lado, el software recomendado de mantenimiento (para Windows) es el siguiente:

- Editor de código java: Eclipse classic 4.2.0 (http://www.eclipse.org/downloads/packages/eclipse-classic-42/junor)
  Configuración de los proyectos del código java del STSGV recomendada: el jre recomendado que sea el que viene con el paquete de instalación del STSGV, el que hay en el directorio software/jdk1.6.0_24/jre
- Editor avanzado de texto: Notepad ++ 5.9.6.2 (http://notepad-plus-plus.org/) Permite ver la codificación y los caracteres retorno de carro.
- Editor de xml: Exchanger XML Editor 3.3.01 (http://www.exchangerxml.com).
  Modo de visionado recomendado: menú Views – Document Views - Viewer, menú View – Collapse All.
- Si el equipo de mantenimiento es Windows, ya sea Windows 7 o Windows XP, se recomienda la instalación de la utilidad “Open command window here” (para el caso de Windows 7) o el plugin “Cmdhere” de las “powertoys” (para el caso de Windows XP). Esta utilidad facilita abrir una ventana de comandos pulsando el botón secundario del ratón sobre la carpeta deseada.
- Editor hexadecimal: Frhed – Free Hex Editor 1.6.0 (http://frhed.sourceforge.net/) Permite ver los xml en formato hexadecimal, para comprobar que los xml estén codificados de manera correcta, es decir como UTF-8 sin BOM (si tiene el BOM entonces el archivo comienza por la cadena en hexadecimal EFBBBF). Este BOM produce errores al firmar los archivos xml.
- Creación de almacenes de claves y certificados: Openssl 0.9.8.
- Editor de diagramas: Dia 0.97.2.
  http://live.gnome.org/Dia

Directrices generales de compilación del código del proyecto

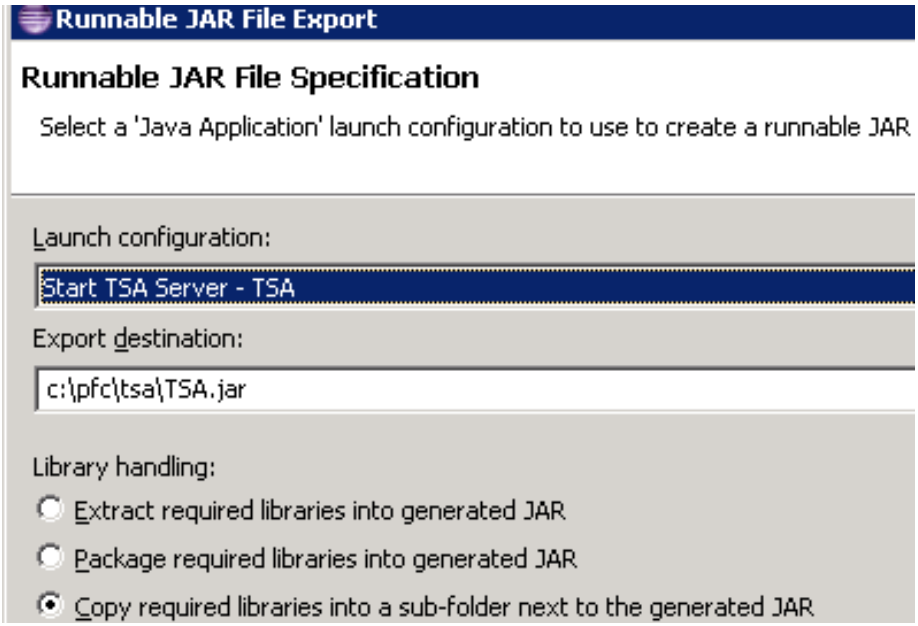
El software Eclipse posee la utilidad de recompilar el código con cada modificación guardada en el editor, de tal manera que para todo el código los archivos compilados (.class) estarán siempre actualizados. Una vez tenemos el código compilado en archivos .class, tenemos que desplegar el código mediante la generación de los empaquetados .jar que contienen los archivos ejecutables de cada módulo.

Como se puede ver en la siguiente ilustración, cada proyecto de código de los tres principales (CERTILOCSIM,STSS,TSA) se convierte en un .jar y una carpeta cuyo nombre acaba en _lib. El .jar contiene todo el código del proyecto correspondiente más aquellas clases del proyecto COMMON de las que dependa. La carpeta "lib" contiene aquellas librerías (.jar) de terceros de las que el proyecto dependa.



En eclipse deberá estar configurada la dependencia de los tres proyectos principales del proyecto común COMMON, esto se define en el "build path" de cada proyecto, en la pestaña "proyects", la ilustración siguiente muestra la dependencia de COMMON del proyecto CERTILOCSIM:

Para generar los .jar junto con la carpeta lib correspondiente, hay que situarse con el botón secundario en el nombre del proyecto en la ventana del “package explorer” y pulsar Export… Seleccionar Java-Runnable Jar File y seleccionar la run configuration correspondiente y donde queremos que nos genere el .jar, y chequear la casilla “Copy required libraries into a sub-folder next to the generated JAR”:



Propiedades configurables de CERTILOC

PROPIEDAD	VALOR	SIGNIFICADO
CERT.CERTILOCSIM.MAIN	CERTILOC.Uc3m.es.crt	Certificado de clave pública de CERTILOC. Este parámetro se usa para la utilidad especial de firmado del Simulador de CERTILOC.
CERT.ROOTCA	Uc3mCA.root.crt	Certificado de clave pública de la Autoridad de Certificacion emisora del certificado CERTILOCSIM.CERT.MAIN
CERTILOCSIM.FILES.DUMMYLOCATIONSFILE	dummyLocations.txt	Fichero de texto con varias localizaciones gps aleatorias.
CERTILOCSIM.SERVER.HOSTNAME	localhost	Nombre o dirección IP del servidor CERTILOC. Este texto es meramente orientativo y no afectará al funcionamiento del programa, podemos dejar el valor localhost (estamos ejecutando en la máquina actual) o bien podemos poner el nombre DNS de el equipo donde esté instalado CERTILOC para que sea más informativo.
CERTILOCSIM.SERVER.PORT	9090	Puerto de escucha del servidor CERTILOC.
CERTILOCSIM.SERVER.ROOTDIRECTORYFORSERVINGFILES	.\\	Directorio raíz del servidor HTTP CERTILOC. Por defecto es el directorio donde se encuentra el archivo .jar ejecutable de CERTILOC.\\, pero podemos poner una ruta relativa al mismo, teniendo cuidado de poner dos caracteres barra invertida.
CERTILOCSIM.TIME.DEFAULTNTPPOOL	Hora.roa.es	Servidor NTP de donde obtendrá la información temporal
CERTILOCSIM.TIME.ENABLENTPRETRIEVAL	True	Variable booleana que indica si queremos que CERTILOC funcione en modo “obtención de la fecha mediante NTP”
CERTILOCSIM.TIME.FORCEDATE	false	Variable booleana que indica si queremos que CERTILOC funcione en modo “forzar una determinada fecha”. Si es verdadero, la fecha devuelta por la TSA será la determinada por las propiedades cuyo nombre comienza por “CERTILOCSIM.TIME.FORCEDATE”. Si es falso, CERTILOC devolverá la fecha de su sistema.
CERTILOCSIM.TIME.FORCEDATE.DAY	18	Indica el día del mes a devolver para la fecha si CERTILOC funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si CERTILOCSIM.TIME.FORCEDATE es verdadero.
CERTILOCSIM.TIME.FORCEDATE.HOUR24H	18	Indica la hora del día, formato 24H a devolver para la

		fecha si CERTILOC funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si CERTILOCSIM.TIME.FORCEDATE es verdadero.
CERTILOCSIM.TIME.FORCEDATE.MICROSECOND	0	Indica el número de microsegundos de tiempo a devolver para la fecha si CERTILOC funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si CERTILOCSIM.TIME.FORCEDATE es verdadero.
CERTILOCSIM.TIME.FORCEDATE.MILLISECOND	350	Indica el número de milisegundos de tiempo a devolver para la fecha si CERTILOC funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si CERTILOCSIM.TIME.FORCEDATE es verdadero.
CERTILOCSIM.TIME.FORCEDATE.MINUTE	45	Indica el minuto de tiempo a devolver para la fecha si CERTILOC funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si CERTILOCSIM.TIME.FORCEDATE es verdadero.
CERTILOCSIM.TIME.FORCEDATE.MONTH	10	Indica el mes de la fecha a devolver para la fecha si l CERTILOC funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si CERTILOCSIM.TIME.FORCEDATE es verdadero.
CERTILOCSIM.TIME.FORCEDATE.SECOND	30	Indica el segundo de tiempo a devolver para la fecha si l CERTILOC funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si CERTILOCSIM.TIME.FORCEDATE es verdadero.
CERTILOCSIM.TIME.FORCEDATE.TIMEZONE	GMT+02:00	Indica la zona horaria de la fecha a devolver para la fecha si CERTILOC funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si CERTILOCSIM.TIME.FORCEDATE es verdadero.
CERTILOCSIM.TIME.FORCEDATE.YEAR	2012	Indica el año de la fecha a devolver para la fecha si CERTILOC funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si CERTILOCSIM.TIME.FORCEDATE es verdadero.
FILES.DEFAULTSIGNEDSPATIALTEMPORALCERTIFICATE.FILENAME	signedSTC.xml	Nombre por defecto de los Certificados Espacio Temporales servidos por el Simulador de CERTILOC.
FILES.DEFAULTTOBESIGNEDSPATIALTEMPORALCERTIFICATE.FILENAME	unsignedSTC.xml	Nombre por defecto del Certificado Espacio Temporal a firmar mediante la utilidad especial de firmado de CERTILOC.
KEYSTORE.CERTILOCSIM.FILENAME	CERTILOCKeystore.jks	Nombre del fichero almacén de claves del Simulador de CERTILOC. Este parámetro se usa para la utilidad especial de firmado del Simulador de CERTILOC.
KEYSTORE.CERTILOCSIM.KEYALIAS	1	Alias de la clave privada de CERTILOC dentro del almacén de claves. Este parámetro se usa para la utilidad especial de firmado del Simulador de CERTILOC.

KEYSTORE.CERTILOCSIM.KEYPASS	las rosas son rojas	Clave de acceso al almacén de claves.Este parámetro se usa para la utilidad especial de firmado del Simulador de CERTILOC.
KEYSTORE.CERTILOCSIM.STOREPASS	las rosas son rojas	Clave de acceso a la clave privada de CERTILOC dentro del almacén de claves. Este parámetro se usa para la utilidad especial de firmado del Simulador de CERTILOC.
PROPIEDAD	VALOR	SIGNIFICADO
NS.DS.PREFIX	ds	Prefijo para el espacio de nombres de XML Signature.
NS.DS.URI	http://www.w3.org/2000/09/xmldsig#	Declaración del espacio de nombres de XML Signature.
NS.GML.PREFIX	gml	Prefijo para el espacio de nombres de gml.
NS.GML.URI	http://www.opengis.net/gml	Declaración del espacio de nombres de gml.
NS.SAML.PREFIX	saml	Prefijo para el espacio de nombres del lenguaje SAML.
NS.SAML.URI	urn:oasis:names:tc:SAML:2.0:assertion	Declaración del espacio de nombres del lenguaje SAML.
NS.STA.PREFIX	Sta	Prefijo para el espacio de nombres de STA.
NS.STA.URI	http://sta	Declaración del espacio de nombres de STA.
NS.TSP.PREFIX	tsp	Prefijo para el espacio de nombres de TSP (Timestamping).
NS.TSP.URI	http://www.esat.kuleuven.ac.be/~kwouters/2002/08/xmltsp#	Declaración del espacio de nombres de TSP (Timestamping)..
NS.XADES.PREFIX	xades	Prefijo para el espacio de nombres de XADES
NS.XADES.URI	http://uri.etsi.org/01903/v1.1.1#	Declaración del espacio de nombres de XADES
NS.XS.PREFIX	xs	Prefijo para el espacio de nombres de XML Schema
NS.XS.URI	http://www.w3.org/2001/XMLSchema	Declaración del espacio de nombres de XML Schema
NS.XSI.PREFIX	xsi	Prefijo para el espacio de nombres de XML Schema Instance
NS.XSI.URI	http://www.w3.org/2001/XMLSchema-instance	Declaración del espacio de nombres de XML Schema Instance

Propiedades configurables de la TSA

PROPIEDAD	VALOR	SIGNIFICADO
CERT.TSA.MAIN	TSA.Uc3m.es.crt	Nombre del archivo de clave pública de la TSA.
CERT.TSA.INTERMEDIATECA	DtoTelematicaCA.crt	Nombre del archivo de clave pública de la Autoridad de Certificación intermedia emisora del certificado de la TSA definido en CERT.TSA.MAIN
CERT.ROOTCA	Uc3mCA.root.crt	Nombre del archivo de clave pública de la Autoridad de Certificación raíz emisora del certificado de la Autoridad de Certificación Intermedia definido en CERT.TSA.INTERMEDIATECA.
FILES.TIMESTAMPREQUEST.FILENAME	TimeStampRequest.xml	Nombre de archivo del xml de petición de Timestamp a la TSA
FILES.TIMESTAMPRESPONSE.FILENAME	TimeStampResponse.xml	Nombre de archivo del xml de respuesta de Timestamp desde la TSA
KEYSTORE.TSA.FILENAME	TsaKeyStore.jks	Nombre del fichero almacén de claves del servidor de la TSA. Este parámetro se usa para que la TSA firme con su clave privada la respuesta de Sello de Tiempo
KEYSTORE.TSA.KEYALIAS	1	Alias de la clave privada de la TSA dentro del almacén de claves. Este parámetro se usa para que la TSA firme con su clave privada la respuesta de Sello de Tiempo
KEYSTORE.TSA.KEYPASS	las rosas son rojas	Clave de acceso al almacén de claves de la TSA.
KEYSTORE.TSA.STOREPASS	las rosas son rojas	Clave de acceso a la clave privada de la TSA dentro del almacén de claves.
NS.DS.PREFIX	ds	Prefijo para el espacio de nombres de XML Signature.
NS.DS.URI	http://www.w3.org/2000/09/xmldsig#	Declaración del espacio de nombres de XML Signature.
NS.GML.PREFIX	gml	Prefijo para el espacio de nombres de gml.
NS.GML.URI	http://www.opengis.net/gml	Declaración del espacio de nombres de gml.
NS.SAML.PREFIX	saml	Prefijo para el espacio de nombres del lenguaje SAML.
NS.SAML.URI	urn:oasis:names:tc:SAML:2.0:assertion	Declaración del espacio de nombres del lenguaje SAML.
NS.STA.PREFIX	Sta	Prefijo para el espacio de nombres de STA.
NS.STA.URI	http://sta	Declaración del espacio de nombres de STA.
NS.TSP.PREFIX	tsp	Prefijo para el espacio de nombres de TSP (Timestamping).
NS.TSP.URI	http://www.esat.kuleuven.ac.be/~kwouters/2002/08/xmltsp#	Declaración del espacio de nombres de TSP (Timestamping)..
NS.XADES.PREFIX	xades	Prefijo para el espacio de nombres de XADES
NS.XADES.URI	http://uri.etsi.org/01903/v1.1.1#	Declaración del espacio de nombres de XADES
NS.XS.PREFIX	xs	Prefijo para el espacio de nombres de XML Schema

NS.XS.URI	http://www.w3.org/2001/XMLSchema	Declaración del espacio de nombres de XML Schema
NS.XSI.PREFIX	xsi	Prefijo para el espacio de nombres de XML Schema Instance
NS.XSI.URI	http://www.w3.org/2001/XMLSchema-instance	Declaración del espacio de nombres de XML Schema Instance
TIMESTAMPRESPONSE.DEFAULTMAJORSTATUS.CODE	0	Código de estado por defecto de la respuesta de la TSA.
TIMESTAMPRESPONSE.DEFAULTMAJORSTATUS.TEXT	Issued TimeStamp OK	Mensaje de respuesta por defecto de la TSA. El valor por defecto "Issued TimeStamp OK" significa que el Sello de Tiempo se ha emitido correctamente. Texto libre, no afecta al funcionamiento del programa.
TIMESTAMPRESPONSE.DEFAULTTSTINFO.ID	tstInfoElement	Identificador del nodo TSTINFO. Texto libre, no afecta al funcionamiento del programa.
TSA.SERIALNUMBERS.FILENAME	IssuedSerialNumbers.log	Nombre del archivo donde vamos dejando constancia de los serial numbers entregados por la TSA, uno por cada TimeStampResponse entregado. Es un fichero incremental cuyo valor inicial es una línea con un 0 (para que al servir el primer TimeStampResponse este tenga un serial number = 1)
TSA.SERVER.DELETEFILESWHENSERVED	False	Variable booleana que indica si queremos que al terminar de servir una respuesta de Timestamp (TimeStampResponse) que borre los ficheros generados en local (TimeStampRequest.xml y TimeStampResponse.xml)
TSA.SERVER.HOSTNAME	localhost	Nombre o dirección IP del servidor TSA. Este texto es meramente orientativo y no afectará al funcionamiento del programa, podemos dejar el valor localhost (estamos ejecutando en la máquina actual) o bien podemos poner el nombre DNS de el equipo donde esté instalada la TSA para que sea más informativo.
TSA.SERVER.PORT	8080	Puerto de escucha del servidor TSA.
TSA.SERVER.ROOTDIRECTORYFORSERVINGFILES	.\\	Directorio raíz del servidor HTTP TSA. Por defecto es el directorio donde se encuentra el archivo .jar ejecutable de TSA (C:\TSA\).
TSA.TIME.DEFAULTNTPPOOL	europe.pool.ntp.org	Nombre DNS o dirección IP del servidor NTP con el que la TSA conecta para obtener una fecha confiable para devolver en la TimeStampResponse. Este valor solo se usa si TSA.TIME.ENABLENTPRETRIEVAL es verdadero. Nota: se sugiere también como posible valor hora.roa.es.
TSA.TIME.DEFAULTORDERING	true	Valor por defecto del campo Ordering de la respuesta de la TSA
TSA.TIME.DEFAULTTSA.NAME	(FICTIONAL DATA) Uc3m TSA Stratum 3	Nombre por defecto de la TSA. Meramente informativo.
TSA.TIME.DEFAULTTSA.URI	(FICTIONAL DATA) XMLTSP://tsa.uc3m.es	URL por defecto de la TSA. Meramente informativo.
TSA.TIME.ENABLENTPRETRIEVAL	false	Variable booleana que indica si queremos que la TSA funcione en modo "obtención de la fecha mediante NTP"

PROPIEDAD	VALOR	SIGNIFICADO
TSA.TIME.FORCEDATE	false	Variable booleana que indica si queremos que la TSA funcione en modo “forzar una determinada fecha”. Si es verdadero, la fecha devuelta por la TSA será la determinada por las propiedades cuyo nombre comienza por “TSA.TIME.FORCEDATE”. Si es falso, la TSA devolverá la fecha de su sistema.
TSA.TIME.FORCEDATE.DAY	1	Indica el día del mes a devolver para la fecha si la TSA funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si TSA.TIME.FORCEDATE es verdadero.
TSA.TIME.FORCEDATE.HOUR24H	18	Indica la hora del día, formato 24H a devolver para la fecha si la TSA funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si TSA.TIME.FORCEDATE es verdadero.
TSA.TIME.FORCEDATE.MICROSECOND	0	Indica el número de microsegundos de tiempo a devolver para la fecha si la TSA funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si TSA.TIME.FORCEDATE es verdadero.
TSA.TIME.FORCEDATE.MILLISECOND	359	Indica el número de milisegundos de tiempo a devolver para la fecha si la TSA funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si TSA.TIME.FORCEDATE es verdadero.
TSA.TIME.FORCEDATE.MINUTE	45	Indica el minuto de tiempo a devolver para la fecha si la TSA funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si TSA.TIME.FORCEDATE es verdadero.
TSA.TIME.FORCEDATE.MONTH	3	Indica el mes de la fecha a devolver para la fecha si la TSA funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si TSA.TIME.FORCEDATE es verdadero.
TSA.TIME.FORCEDATE.SECOND	30	Indica el segundo de tiempo a devolver para la fecha si la TSA funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si TSA.TIME.FORCEDATE es verdadero.
TSA.TIME.FORCEDATE.YEAR	2008	Indica el año de la fecha a devolver para la fecha si la TSA funciona en modo “forzar una determinada fecha”. Solo se usa esta propiedad si TSA.TIME.FORCEDATE es verdadero.

Propiedades configurables del Sujeto

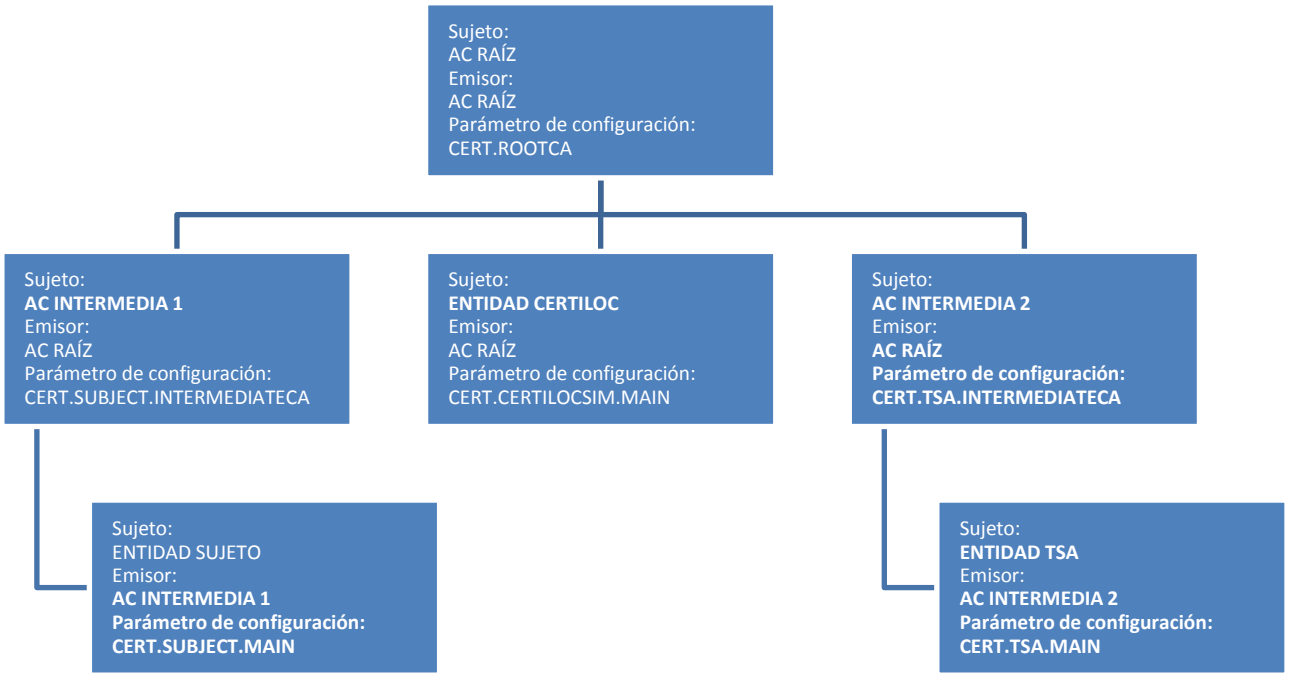
PROPIEDAD	VALOR	SIGNIFICADO
CERT.SUBJECT.MAIN	Subject.crt	Nombre del archivo de clave pública del Sujeto.
CERT.SUBJECT.INTERMEDIATECA	DtoInformaticaCA.crt	Nombre del archivo de clave pública de la Autoridad de Certificación intermedia emisora del certificado del sujeto definido en CERT.SUBJECT.MAIN.
CERT.ROOTCA	Uc3mCA.root.crt	Nombre del archivo de clave pública de la Autoridad de Certificación raíz emisora del certificado de la Autoridad de Certificación Intermedia definido en CERT.SUBJECT.INTERMEDIATECA.
CERTILOCSIM.SERVER.HOSTNAME	localhost	Nombre dns o dirección IP del servidor CERTILOC.
CERTILOCSIM.SERVER.PORT	9090	Puerto de escucha del servidor CERTILOC. Este valor debe ser igual al valor de la propiedad CERTILOCSIM.SERVER.PORT en el directorio por defecto del servidor CERTILOC
FILES.DEFAULTMESSAGEFILE.FILENAME	loren_ipsum.txt	Nombre del archivo de Mensaje por defecto.
FILES.DEFAULTSIGNEDSPATIALTEMPORALCERTIFICATE.FILENAME	signedSTC.xml	Nombre del archivo de Certificado Espacio Temporal por defecto.
FILES.SIGMA.FILENAME	Sigma.xml	Nombre del archivo Sigma
FILES.SPATIALTEMPORALSTAMP.FILENAME	SpatialTemporalStamp.xml	Nombre del archivo de Sello Espacio Temporal
FILES.TIMESTAMPREQUEST.FILENAME	TimeStampRequest.xml	Nombre de archivo del xml de petición de Timestamp a la TSA
FILES.TIMESTAMPRESPONSE.FILENAME	TimeStampResponse.xml	Nombre de archivo del xml de respuesta de Timestamp desde la TSA
KEYSTORE.SUBJECT.FILENAME	SubjectKeystore.jks	Nombre del fichero almacén de claves del STSGV. Este parámetro se usa para que el Sujeto firme con su clave privada la petición de Sello de Tiempo a la TSA
KEYSTORE.SUBJECT.KEYALIAS	1	Alias de la clave privada del STSGV dentro del almacén de claves.
KEYSTORE.SUBJECT.KEYPASS	las rosas son rojas	Clave de acceso al almacén de claves del STSGV.
KEYSTORE.SUBJECT.STOREPASS	las rosas son rojas	Clave de acceso a la clave privada del STSGVdentro del almacén de claves.
NS.DS.PREFIX	ds	Prefijo para el espacio de nombres de XML Signature.
NS.DS.URI	http://www.w3.org/2000/09/xmldsig#	Declaración del espacio de nombres de XML Signature.
NS.GML.PREFIX	gml	Prefijo para el espacio de nombres de gml.
NS.GML.URI	http://www.opengis.net/gml	Declaración del espacio de nombres de gml.
NS.SAML.PREFIX	saml	Prefijo para el espacio de nombres del lenguaje SAML.

NS.SAML.URI	urn:oasis:names:tc:SAML:2.0:assertion	Declaración del espacio de nombres del lenguaje SAML.
NS.STA.PREFIX	Sta	Prefijo para el espacio de nombres de STA.
NS.STA.URI	http://sta	Declaración del espacio de nombres de STA.
NS.TSP.PREFIX	tsp	Prefijo para el espacio de nombres de TSP (Timestamping).
PROPIEDAD	VALOR	SIGNIFICADO
NS.TSP.URI	http://www.esat.kuleuven.ac.be/~kwouters/2002/08/xmltsp#	Declaración del espacio de nombres de TSP (Timestamping)..
NS.XADES.PREFIX	xades	Prefijo para el espacio de nombres de XADES
NS.XADES.URI	http://uri.etsi.org/01903/v1.1.1#	Declaración del espacio de nombres de XADES
NS.XS.PREFIX	xs	Prefijo para el espacio de nombres de XML Schema
NS.XS.URI	http://www.w3.org/2001/XMLSchema	Declaración del espacio de nombres de XML Schema
NS.XSI.PREFIX	xsi	Prefijo para el espacio de nombres de XML Schema Instance
NS.XSI.URI	http://www.w3.org/2001/XMLSchema-instance	Declaración del espacio de nombres de XML Schema Instance
SUBJECT.CONSOLEUI.MODE.ADVANCEDUI	True	Variable que indica si queremos que se muestre el menú de Administrador (true) o por el contrario queremos el menú de usuario final (false)
SUBJECT.CONSOLEUI.ASKFORCERTPASSWORD	False	Indica si queremos que se solicite la password de acceso tanto al almacén de claves como a la clave privada para la firma de Sigma por parte del Sujeto.
TIMESTAMPREQUEST.DEFAULTCERTREQ	True	Valor que deberá tomar la variable CertReq en las TimeStampRequest. Deberá ser true o false
TIMESTAMPREQUEST.DEFAULTDOCUMENTATIONREFERENCE1	STS User Manual	Texto variable con una referencia a incluir para la policy de la TimeStampRequest. Texto libre, no afecta al funcionamiento del programa.
TIMESTAMPREQUEST.DEFAULTDOCUMENTATIONREFERENCE2	STS Technical Specifications	Texto variable con una referencia a incluir para la policy de la TimeStampRequest. Texto libre, no afecta al funcionamiento del programa.
TIMESTAMPREQUEST.DEFAULTID	DefaultTimeStampRequestId	Identificador por defecto de la TimeStampRequest. Texto libre, no afecta al funcionamiento del programa.
TIMESTAMPREQUEST.DEFAULTMESSAGEIMPRINTSID	Hash to send to TSA	Identificador del Message Imprints. Texto libre, no afecta al funcionamiento del programa.
TIMESTAMPREQUEST.DEFAULTMESSAGENODE.ID	Message	Id para poner al nodo File donde irá el resumen del fichero de Mensaje.
TIMESTAMPREQUEST.DEFAULTSPIDESCRIPTION	This is the policy for the STS system.	Descripción de la policy de la TimeStampRequest. Texto libre, no afecta al funcionamiento del programa.
TIMESTAMPREQUEST.DEFAULTSPIDIGEST		Hash de la policy. Como no tenemos definida una formalmente, se ha puesto un digest arbitrario. Texto libre, no afecta al funcionamiento del programa.
TIMESTAMPREQUEST.DEFAULTSPIIDENTIFIER	(FICTIONAL DATA)	Nombre de la policy. Texto libre, no afecta al funcionamiento del

	http://www.uc3m.es/DtoInformatica/DocInterna/STSPolicy.odt	programa.
TIMESTAMPREQUEST.DEFAULTSTCNODE.ID	CERTILOC STC	Id para poner al nodo Certificate donde irá el resumen del fichero Certificado Espacio Temporal.
TIMESTAMPREQUEST.DEFAULTTYPE	DefaultTimeStampRequestType	Tipo de TimeStampRequest. Texto libre, no afecta al funcionamiento del programa.
PROPIEDAD	VALOR	SIGNIFICADO
TIMESTAMPREQUEST.QUALIFIERTYPE.OIDASURI	OIDAsURI	Tipo de identificador definido por una URI.
TIMESTAMPREQUEST.QUALIFIERTYPE.OIDASURN	OIDAsURN	Tipo de identificador definido por una URN
TSA.SERVER.HOSTNAME	localhost	Nombre dns o ip de la máquina que alberga el servidor de la TSA. Deberá ser la dirección o nombre correcto y que la máquina de la TSA sea accesible a nivel de red.
TSA.SERVER.PORT	8080	Puerto de escucha de la TSA a donde enviar las peticiones de Sello de Tiempo (TimeStamp Requests)

Mantenimiento de la estructura de certificación

Este software se entrega para seguir la estructura de certificación siguiente: una entidad raíz que tiene como hijas dos entidades intermedia y una entidad para certiloc, y como hijas de las entidades intermedias los certificados de la TSA y el Sujeto, como se puede ver en la siguiente figura:



La

La estructura mínima posible debe ser, por motivos obvios, tres certificados: 1) Certificado de Sujeto, 2) Certificado de TSA y 3) Certificado de CERTILOC.

Para cambiar esta estructura, sería necesario hacer cambios a los siguientes niveles:

- A nivel de parámetros de configuración: añadir o quitar los parámetros necesarios.

  Por ejemplo, si queremos añadir una AC intermedia por encima de la AC INTERMEDIA 2, una posible sugerencia es seguir estos pasos: 1) renombrar el parámetro CERT.TSA.INTERMEDIATECA a CERT.TSA.INTERMEDIATECA.1 2) añadir un parámetro cuyo nombre podría ser CERT.TSA.INTERMEDIATECA.2. Como ilustra la siguiente figura:

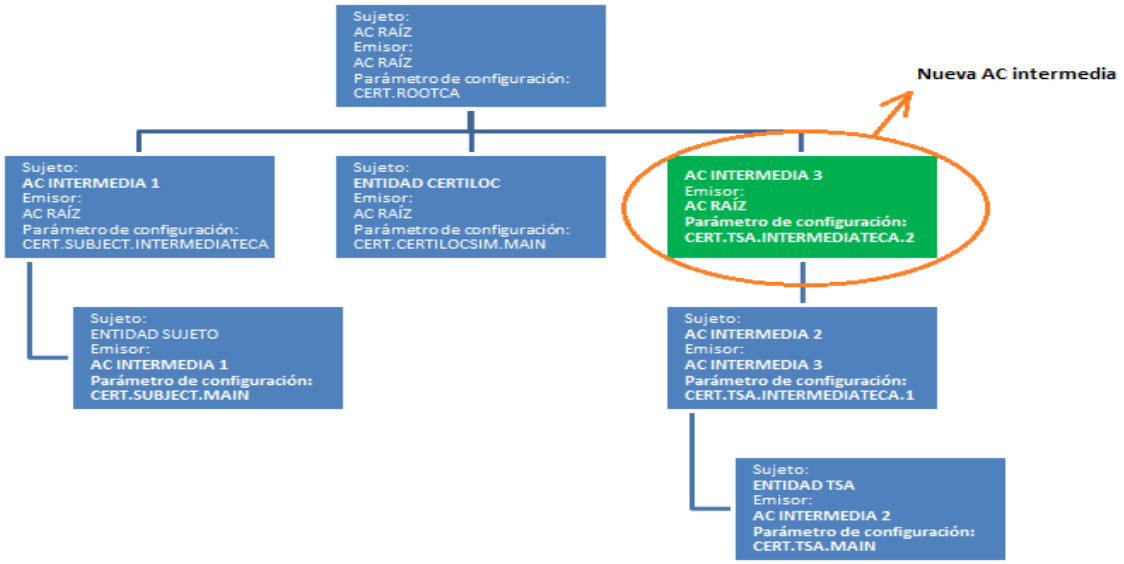


Figura: Adición de una entidad intermedia

- A nivel de código:

  Para el caso del ejemplo se añade la línea para incluir la nueva entidad intermedia al certificate path de la TSA:

```
String sTsaCertFile = Configuration.getInstance().getProperty("CERT.TSAMAIN");
String sTsaIntermediateCertFile  = Configuration.getInstance().getProperty("CERT.TSAINTERMEDIATECA");
String sTsaIntermediate2CertFile = Configuration.getInstance().getProperty("CERT.TSAINTERMEDIATECA.2");
String sTsaRootCertFile  = Configuration.getInstance().getProperty("CERT.ROOTCA");
File fMainCertFile = new File(sTsaCertFile);
File fIntermediateCertFile = new File(sTsaIntermediateCertFile);
File fIntermediateCertFile2 = new File(sTsaIntermediate2CertFile);
File fRootCertFile = new File(sTsaRootCertFile);

if  (fMainCertFile.isFile() && fMainCertFile.canRead() &&
        fIntermediateCertFile.isFile() && fIntermediateCertFile.canRead() &&
        fIntermediateCertFile2.isFile() && fIntermediateCertFile2.canRead() &&
        fRootCertFile.isFile() && fRootCertFile.canRead()  )
{

    certificatePathUrls.add(sTsaCertFile);
    certificatePathUrls.add(sTsaIntermediateCertFile);
    certificatePathUrls.add(sTsaIntermediate2CertFile);
    certificatePathUrls.add(sTsaRootCertFile);
```

- A nivel de certificados:

  Utilizar la herramienta openssl para crear una nueva autoridad de certificación intermedia con la configuración adecuada. Habrá que crear de nuevo el certificado de la autoridad de certificación intermedia 3 para que su emisor sea la nueva autoridad de certificación (ver Figura: adición de una entidad intermedia) . Habrá que revisar también el valor de las restricciones básicas – propiedad “longitud de la cadena de certificación” para que se adecúe a la nueva situación.